Министерство науки и высшего образования Российской Федерации
Иркутский национальный исследовательский технический университет

Факультет среднего профессионального образования
Машиностроительный колледж

А.Д. Шипилова

ОУП.05 ИНФОРМАТИКА

Методические указания
по выполнению практических работ

Издательство
Иркутского национального исследовательского технического университета
2025

Рекомендовано к изданию Учебно-методической комиссией факультета среднего профессионального образования

Автор
Преподаватель машиностроительного колледжа факультета среднего профессионального образования ФГБОУ ВО «ИРНИТУ» **А.Д. Шипилова**

Шипилова А.Д. ОУП.05п Информатика: метод. указания по выполнению практических, лабораторных и самостоятельных работ. – Иркутск: Изд-во ИРНИТУ, 2025. – 244с.

Соответствуют требованиям ФГОС среднего профессионального образования по специальности 23.02.07 «Техническое обслуживание и ремонт автотранспортных средств».

Предназначены для студентов машиностроительного колледжа, изучающих предмет «Информатика» в рамках подготовки специалистов среднего звена.

Введение

Методические указания составлены в соответствии с Федеральным государственным образовательным стандартом среднего профессионального образования по специальности 23.02.07 «Техническое обслуживание и ремонт автотранспортных средств», Федеральным государственным образовательным стандартом среднего общего образования.

Цель методических указаний является: определенный этап сформированности следующих общих и профессиональных компетенций:

Код и наименование формируемых компетенций	**Планируемые результаты освоения предмета**	
	Общие	**Предметные**
ОК 01 Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам	самостоятельно формулировать и актуализировать проблему, рассматривать её всесторонне; устанавливать существенный признак или основания для сравнения, классификации и обобщения; определять цели деятельности, задавать параметры и критерии их достижения; выявлять закономерности и противоречия в рассматриваемых явлениях; разрабатывать план решения проблемы с учётом анализа имеющихся материальных и нематериальных ресурсов; вносить коррективы в деятельность, оценивать соответствие результатов целям, оценивать риски последствий деятельности; координировать и выполнять работу в условиях реального, виртуального и комбинированного взаимодействия; развивать креативное мышление при решении жизненных проблем.	умение характеризовать большие данные, приводить примеры источников их получения и направления использования, умение классифицировать основные задачи анализа данных (прогнозирование, классификация, кластеризация, анализ отклонений), понимать последовательность решения задач анализа данных: сбор первичных данных, очистка и оценка качества данных, выбор и/или построение модели, преобразование данных, визуализация данных, интерпретация результатов; понимание основных принципов устройства и функционирования современных стационарных и мобильных компьютеров, тенденций развития компьютерных технологий; умение использовать при решении задач свойства позиционной записи чисел, алгоритма построения записи числа в позиционной системе счисления с заданным основанием и построения числа по строке, содержащей запись этого числа в позиционной системе счисления с заданным

		основанием, умение выполнять арифметические операции в позиционных системах счисления; умение выполнять преобразования логических выражений, используя законы алгебры логики, умение строить логическое выражение в дизъюнктивной и конъюнктивной нормальных формах по заданной таблице истинности, исследовать область истинности высказывания, содержащего переменные, решать несложные логические уравнения и системы уравнений;
ОК 02 Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности	владеть навыками учебно-исследовательской и проектной деятельности, навыками разрешения проблем, способностью и готовностью к самостоятельному поиску методов решения практических задач, применению различных методов познания; овладеть видами деятельности по получению нового знания, его интерпретации, преобразованию и применению в различных учебных ситуациях, в том числе при создании учебных и социальных проектов; формирование научного типа мышления, владение научной терминологией, ключевыми понятиями и методами; ставить и формулировать собственные задачи в образовательной деятельности и жизненных ситуациях; выявлять причинно-следственные связи и актуализировать задачу, выдвигать гипотезу её решения, находить аргументы для доказательства своих утверждений, задавать	умение создавать структурированные текстовые документы и демонстрационные материалы с использованием возможностей современных программных средств и облачных сервисов; умение использовать электронные таблицы для анализа, представления и обработки данных (включая вычисление суммы, среднего арифметического, наибольшего и наименьшего значений, решение уравнений, выбор оптимального решения, подбор линии тренда, решение задач прогнозирования). умение строить неравномерные коды, допускающие однозначное декодирование сообщений (префиксные коды), использовать простейшие коды, которые позволяют обнаруживать и исправлять ошибки при передаче данных, строить код, обеспечивающий наименьшую возможную среднюю длину сообщения при известной частоте символов,

	параметры и критерии решения; анализировать полученные в ходе решения задачи результаты, критически оценивать их достоверность, прогнозировать изменение в новых условиях; давать оценку новым ситуациям, оценивать приобретённый опыт; осуществлять целенаправленный поиск переноса средств и способов действия в профессиональную среду; переносить знания в познавательную и практическую области жизнедеятельности; интегрировать знания из разных предметных областей; выдвигать новые идеи, предлагать оригинальные подходы и решения, ставить проблемы и задачи, допускающие альтернативные решения.	пояснять принципы работы простых алгоритмов сжатия данных; умение решать алгоритмические задачи, связанные с анализом графов (задачи построения оптимального пути между вершинами графа, определения количества различных путей между вершинами ориентированного ациклического графа), умение использовать деревья при анализе и построении кодов и для представления арифметических выражений, при решении задач поиска и сортировки, умение строить дерево игры по заданному алгоритму, разрабатывать и обосновывать выигрышную стратегию игры; умение разрабатывать и реализовывать в виде программ базовые алгоритмы, умение использовать в программах данные различных типов с учётом ограничений на диапазон их возможных значений, применять при решении задач структуры данных (списки, словари, стеки, очереди, деревья), использовать базовые операции со структурами данных, применять стандартные и собственные подпрограммы для обработки числовых данных и символьных строк, использовать при разработке программ библиотеки подпрограмм, знать функциональные возможности инструментальных средств среды разработки, умение использовать средства отладки программ в среде программирования, умение документировать программы;

		умение создавать веб-страницы; умение использовать компьютерно-математические модели для анализа объектов и процессов: формулировать цель моделирования, выполнять анализ результатов, полученных в ходе моделирования, оценивать соответствие модели моделируемому объекту или процессу, представлять результаты моделирования в наглядном виде; умение организовывать личное информационное пространство с использованием различных средств цифровых технологий, понимание возможностей цифровых сервисов государственных услуг, цифровых образовательных сервисов; умение критически оценивать информацию, полученную из сети Интернет
ПК 2.4 Осуществлять документооборот и учет движения запасных частей при осуществлении работ по техническому обслуживанию и ремонту автотранспортных средств	- оформлять техническую документацию	Уметь разрабатывать технологическую документацию для проведения работ по техническому обслуживанию и ремонту автотранспортных средств

Общее количество часов на практические работы составляет – 150

Информационное обеспечение:

Перечень основной и дополнительной литературы, электронных ресурсов:

Основная литература:

1. Семакин, И. Г. Информатика. 10 класс. Углубленный уровень: в 2 частях. Часть 1: учебник / И. Г. Семакин, Т. Ю. Шеина, Л. В. Шестакова. - 4-е изд., стер -

Москва: Издательство «Просвещение», 2022. - 208 с. - ISBN 978-5-09-101612-3. - Текст: электронный. - URL: https://znanium.com/catalog/product/2089890 (дата обращения: 22.03.2024). – Режим доступа: по подписке.

2. Семакин, И. Г. Информатика. 11 класс. Углубленный уровень : в 2 частях. Часть 1: учебник / И. Г. Семакин, Е. К. Хеннер, Л. В. Шестакова. - 4-е изд., стер. - Москва: Издательство «Просвещение», 2022. - 176 с. - ISBN 978-5-09-101614-7. - Текст: электронный. - URL: https://znanium.com/catalog/product/2089907 (дата обращения: 22.03.2024). – Режим доступа: по подписке.

Дополнительная литература:

1. Угринович, Н. Д. Информатика. 10-й класс. Базовый уровень: учебник / Н. Д. Угринович. - 5-е изд.,стер. - Москва: Издательство «Просвещение», 2022. - 288 с. - ISBN 978-5-09-101608-6. - Текст: электронный. - URL: https://znanium.com/catalog/product/2089880 (дата обращения: 22.03.2024). – Режим доступа: по подписке.

2. Угринович, Н. Д. Информатика. 11 класс. Базовый уровень: учебник / Н. Д. Угринович. - 4-е изд., стер. - Москва: Издательство «Просвещение», 2022. - 272 с. - ISBN 978-5-09-101609-3. - Текст: электронный. - URL: https://znanium.com/catalog/product/2089883 (дата обращения: 22.03.2024). – Режим доступа: по подписке.

3. Поляков, К. Ю. Информатика. 10 класс. Базовый и углубленный уровни (в двух частях). Часть 1 : учебник / К. Ю. Поляков, Е. А. Еремин. - Москва: Издательство «Просвещение», 2022. - 352 с. - ISBN 978-5-09-099486-6. - Текст : электронный. - URL: https://znanium.ru/catalog/product/1923119 (дата обращения: 22.03.2024). – Режим доступа: по подписке.

4. Поляков, К. Ю. Информатика. 11 класс. Базовый и углубленный уровни. Часть 1: учебник / К. Ю. Поляков, Е. А. Еремин. — 5-е изд., стер. — Москва: Просвещение, 2023. - 238 с. - ISBN 978-5-09-103617-6. - Текст: электронный. - URL: https://znanium.com/catalog/product/2089841 (дата обращения: 22.03.2024). – Режим доступа: по подписке.)

Таблица – Перечень практических работ

№	Тема	Вид, номер и название работы	Коды общих и профессиональных компетенций	Количество часов (*из*
Семестр 1				

1.	Тема 1.2. Подходы к измерению информации.	Практическая работа №1 Дискретизация текстовой, графической и звуковой информации	ОК 02	2
2.	Тема 1.3 Компьютер и цифровое представление информации. Устройство компьютера	Практическая работа №2 Работа с операционной системой, классификация программ, характеристика устройств компьютера.	ОК 02	2
3.	Тема 1.4 Кодирование информации. Системы счисления	Практическая работа №3 Перевод чисел в разные системы счисления	ОК 02	2
4.	Тема 1.5 Элементы комбинаторики, теории множеств и математической логики	Практическая работа №4 Построение и анализ таблиц истинности в табличном процессоре. Построение логических схем и решение логических задач.	ОК 02	2
5.	Тема 1.6 Компьютерные сети: локальные сети, сеть Интернет/ Службы Интернета	Практическая работа №5 Методы поиска в Интернете.	ОК 01 ОК 02	2
6.	Тема 1.7 Сетевое хранение данных и цифрового контента	Практическая работа №6 Организация личного информационного пространства. Разделение прав доступа в облачных хранилищах. Облачные хранилища данных.	ОК 01 ОК 02	2
7.	Тема 1.8 Информационная	Практическая работа №7	ОК 01 ОК 02	2

	безопасность	Антивирусные программы. Программы-архиваторы		
8.	Тема 2.1 Обработка информации в текстовых процессорах	Практическая работа №8 Редактирование и форматирование текстовых документов создание оглавлений. Оформление рефератов. Вёрстка документов с таблицами, рисунками и математическими формулами.	ОК 02	2
9.	Тема 2.2 Технология создания структурированных текстовых документов	Практическая работа №9 Многостраничные документы Практическая работа №10 Коллективная работа с документами	ОК 02	2 2
10.	Тема 2.3 Компьютерная графика и мультимедиа	Практическая работа №11 Обработка цифровых фотографий (кадрирование, исправление перспективы, коррекция уровней, коррекция цвета). Многослойные изображения. Практическая работа №12 Векторная графика	ОК 02	4 4
11.	Тема 2.4 Представление профессиональной информации в виде презентаций	Практическая работа №13 Интерактивные мультимедийные объекты на слайде	ОК 02	4
12.	Тема 2.5 Гипертекстовое	Практическая работа №14	ОК 02	2

	представление информации	Создание веб-страницы включающей текст, мультимедийные объекты (рисунки, звуковые данные, видео). Создание форм на HTML.Практическая работа №15 Оформление страницы с помощью каскадных таблиц стилей.		2
13.	Тема 3.1 Модели и моделирование. Этапы моделирования	Практическая работа №16 Исследование физической модели	ОК 02	4
14.	Тема 3.2 Искусственный интеллект	Практическая работа №17 Средства искусственного интеллекта	ОК 02	4
15.	Тема 3.3 Технология обработки информации в электронных таблицах	Практическая работа №18 Обработка данных. Вычисление суммы, среднего арифметического, наибольшего и наименьшего значений. Сортировка, фильтрация, условное форматирование.	ОК 02	4
16.	Тема 3.4 Формулы и функции в электронных таблицах.	Практическая работа №19 Численное решение уравнений с помощью подбора параметра.	ОК 02	4
17.	Тема 3.5 Визуализация данных в электронных таблицах	Практическая работа №20 Наглядное представление результатов статистической обработки данных в виде диаграмм	ОК 02	6

		средствами редактора электронных таблиц		
18.	Тема 3.6 Моделирование в электронных таблицах (на примерах задач из профессиональной области)	Практическая работа №21 Решение задач оптимизации с помощью электронных таблиц.	ОК 02	4
19.	Тема 3.7 Базы данных как модель предметной области	Практическая работа №22 Разработка многотабличной базы данных.		4
		Итого	ОК 01 ОК 02	66
		Семестр 2		
20.	Тема 3.7 Базы данных как модель предметной области	Практическая работа №23 Запросы к многотабличной базе данных.	ОК 02	4
21.	Тема 4.1 Понятие алгоритма и основные алгоритмические структуры. Введение в язык программирования Python.	Практическая работа №24 Выделение и обработка цифр целого числа в различных системах счисления с использованием операций целочисленной арифметики. Практическая работа №25 Решение задач методом перебора	ОК 02	4 4
22.	Тема 4.2 Основные алгоритмические конструкции на Python	Практическая работа №26 Обработка данных, хранящихся в файлах.	ОК 02	2
23.	Тема 4.3 Вспомогательные алгоритмы	1Практическая работа №27 Использование подпрограмм стандартной библиотеки языка программирования.	ОК 02	4

		Практическая работа №28 Разработка подпрограмм.		4
24.	Тема 4.4 Алгоритмы обработки символьных данных	Практическая работа №29 Посимвольная обработка строк.	ОК 02	2
25.	Тема 4.5 Алгоритмы обработки массивов	Практическая работа №30 Заполнение массива. Практическая работа №31 Простые методы сортировки массива.	ОК 02	4 4
26.	Тема 4.6 Работа со списками, словарями, стеками	Практическая работа №32 Применение списков и словарей в реальных задачах.	ОК 02	4
27.	Тема 4.7 Алгоритмы и структуры данных	Практическая работа №33 Поиск простых чисел в заданном диапазоне. Практическая работа №34 Использование деревьев для вычисления арифметических выражений. Практическая работа №35 Вычисление длины кратчайшего пути между вершинами графа (алгоритм Дейкстры). Практическая работа №36 Подсчёт количества вариантов с помощью динамического программирования.	ОК 02	2 2 2 4
28.	Тема 5.1. Растровая и векторная графика. Форматы изображений,	Практическая работа №37 Использование растровой графики	ОК 02 ПК 2.4	4

	конвертация и оптимизация	для хранения фотографий. Конвертация с целью снижения объёма изображения		
29.	Тема 5.3. Интерфейс GIMP. Многооконный режим, стыкуемые диалоги, однооконный режим. Слои	Практическое занятие №38 Интерфейс и настройка его частей. Однооконный и многооконный режим Практическое занятие №39 Управление диалогами. Окно слоёв изображения	ОК 02 ПК 2.4	4 4
30.	Тема 5.4. Разрешение изображения. Навигация, масштабирование, кадрирование, аффинные преобразования	Практическая работа №40 Масштабирование и кадрирование: основы работы с изображениями в GIMP.	ОК 02 ПК 2.4	4
31.	Тема 5.5 Заливка, фильтры и инструменты рисования	Практическая работа №41 Заливка, фильтры и инструменты рисования: основы работы с изображениями в GIMP	ОК 02 ПК 2.4	4
32.	Тема 5.6 Выделение. Контуры. Комбинирование изображений	Практическая работа 42 Выделение, контуры, комбинирование изображений: основы работы с изображениями в GIMP.	ОК 02 ПК 2.4	4
33.	Тема 5.7 Быстрая маска и преобразование цвета	Практическая работа №43 Создание эффектов с использованием быстрой маски и цветовых преобразований в	ОК 02 ПК 2.4	4

		GIMP		
34.	Тема 5.8 Создание градиентов	Практическая работа №44 Градиенты в GIMP: от создания до применения в графических работах	ОК 02 ПК 2.4	4
35.	Тема 5.9 Создание анимированного изображения в	Практическая работа №45 Создание анимированных GIF в GIMP	ОК 02 ПК 2.4	4
			Итого	84ч

Практическая работа № 1

Тема: Дискретизация текстовой, графической и звуковой информации

Количество часов на выполнение: 2, из них на практическую подготовку - 2ч.

Цель работы: изучить способы представления текстовой, графической, звуковой информации и видеоинформации, научиться записывать информацию в различных кодировках.

Оборудование: Персональный компьютер

Методика выполнения задания
Теоретическая часть

1.Краткие теоретические сведения

Вся информация, которую обрабатывает компьютер, должна быть представлена двоичным кодом с помощью двух цифр 0 и 1. Эти два символа принято называть двоичными цифрами или битами. С помощью двух цифр 0 и 1 можно закодировать любое сообщение. Это явилось причиной того, что в компьютере обязательно должно быть организованно два важных процесса: кодирование и декодирование.

Кодирование – преобразование входной информации в форму, воспринимаемую компьютером, то есть двоичный код.

Декодирование – преобразование данных из двоичного кода в форму, понятную человеку.

С точки зрения технической реализации использование двоичной системы счисления для кодирования информации оказалось намного более простым, чем применение других способов. Действительно, удобно кодировать информацию в виде последовательности нулей и единиц, если представить эти значения как два возможных устойчивых состояния электронного элемента:

- 0 – отсутствие электрического сигнала;
- 1 – наличие электрического сигнала.

Эти состояния легко различать. Недостаток двоичного кодирования – длинные коды. Но в технике легче иметь дело с большим количеством простых элементов, чем с небольшим числом сложных.

Способы кодирования и декодирования информации в компьютере, в первую очередь, зависит от вида информации, а именно, что должно кодироваться: числа, текст, графические изображения или звук.

Аналоговый и дискретный способ кодирования

Человек способен воспринимать и хранить информацию в форме образов (зрительных, звуковых, осязательных, вкусовых и обонятельных). Зрительные образы могут быть сохранены в виде изображений (рисунков, фотографий и так далее), а звуковые - зафиксированы на пластинках, магнитных лентах, лазерных дисках и так далее.

Информация, в том числе графическая и звуковая, может быть представлена в аналоговой или дискретной форме. При аналоговом представлении физическая величина принимает бесконечное множество значений, причем ее значения изменяются непрерывно. При дискретном представлении физическая величина принимает конечное множество значений, причем ее величина изменяется скачкообразно.

Примером аналогового представления графической информации может служить, например, живописное полотно, цвет которого изменяется непрерывно, а дискретного – изображение, напечатанное с помощью струйного принтера и состоящее из отдельных точек разного цвета. Примером аналогового хранения звуковой информации является виниловая пластинка (звуковая дорожка изменяет свою форму непрерывно), а дискретного – аудио компакт-диск (звуковая дорожка которого содержит участки с различной отражающей способностью).

Преобразование графической и звуковой информации из аналоговой формы в дискретную производится путем дискретизации, то есть разбиения непрерывного графического изображения и непрерывного (аналогового) звукового сигнала на отдельные элементы. В процессе дискретизации производится кодирование, то есть присвоение каждому элементу конкретного значения в форме кода.

Дискретизация – это преобразование непрерывных изображений и звука в набор дискретных значений в форме кодов.

Кодирование изображений

Создавать и хранить графические объекты в компьютере можно двумя способами – как растровое или как векторное изображение. Для каждого типа изображений используется свой способ кодирования.

Кодирование растровых изображений

Растровое изображение представляет собой совокупность точек (пикселей) разных цветов. **Пиксель** – минимальный участок изображения, цвет которого можно задать независимым образом.

В процессе кодирования изображения производится его пространственная дискретизация. Пространственную дискретизацию изображения можно сравнить с построением изображения из мозаики (большого количества маленьких разноцветных стекол). Изображение разбивается на отдельные маленькие фрагменты (точки), причем каждому фрагменту присваивается значение его цвета, то есть код цвета (красный, зеленый, синий и так далее).

Для черно-белого изображения информационный объем одной точки равен одному биту (либо черная, либо белая – либо 1, либо 0).

Для четырех цветного – 2 бита.

Для 8 цветов необходимо – 3 бита.

Для 16 цветов – 4 бита.

Для 256 цветов – 8 бит (1 байт).

Качество изображения зависит от количества точек (чем меньше размер точки и, соответственно, больше их количество, тем лучше качество) и количества используемых цветов (чем больше цветов, тем качественнее кодируется изображение).

Для представления цвета в виде числового кода используются две обратных друг другу цветовые модели: RGB или CMYK. Модель RGB используется в телевизорах, мониторах, проекторах, сканерах, цифровых фотоаппаратах… Основные цвета в этой модели: красный (Red), зеленый (Green), синий (Blue). Цветовая модель CMYK используется в полиграфии при формировании изображений, предназначенных для печати на бумаге.

Цветные изображения могут иметь различную глубину цвета, которая задается количеством битов, используемых для кодирования цвета точки.

Если кодировать цвет одной точки изображения тремя битами (по одному биту на каждый цвет RGB), то мы получим все восемь различных цветов.

R	G	B	Цвет
1	1	1	Белый
1	1	0	Желтый
1	0	1	Пурпурный
1	0	0	Красный
0	1	1	Голубой
0	1	0	Зеленый
0	0	1	Синий
0	0	0	Черный

На практике же, для сохранения информации о цвете каждой точки цветного изображения в модели RGB обычно отводится 3 байта (то есть 2^4 бита) - по 1 байту (то есть по 8 бит) под значение цвета каждой составляющей. Таким образом, каждая RGB-составляющая может принимать значение в диапазоне от 0 до 255 (всего 2^8=256 значений), а каждая точка изображения, при такой системе кодирования может быть окрашена в один из 16 777 216 цветов. Такой набор цветов принято называть True Color (правдивые цвета), потому что человеческий глаз все равно не в состоянии различить большего разнообразия.

Для того чтобы на экране монитора формировалось изображение, информация о каждой точке (код цвета точки) должна храниться в видеопамяти компьютера. Рассчитаем необходимый объем видеопамяти для одного из графических режимов. В

современных компьютерах разрешение экрана обычно составляет 1280x1024 точек. Т.е. всего 1280 * 1024 = 1310720 точек. При глубине цвета 32 бита на точку необходимый объем видеопамяти:

32 * 1310720 = 41943040 бит = 5242880 байт = 5120 Кб = 5 Мб.

Растровые изображения очень чувствительны к масштабированию (увеличению или уменьшению). При уменьшении растрового изображения несколько соседних точек преобразуются в одну, поэтому теряется различимость мелких деталей изображения. При увеличении изображения увеличивается размер каждой точки и появляется ступенчатый эффект, который можно увидеть невооруженным глазом.

Кодирование векторных изображений

Векторное изображение представляет собой совокупность графических примитивов (точка, отрезок, эллипс…). Каждый примитив описывается математическими формулами. Кодирование зависит от прикладной среды.

Достоинством векторной графики является то, что файлы, хранящие векторные графические изображения, имеют сравнительно небольшой объем.

Важно также, что векторные графические изображения могут быть увеличены или уменьшены без потери качества.

Графические форматы файлов

Форматы графических файлов определяют способ хранения информации в файле (растровый или векторный), а также форму хранения информации (используемый алгоритм сжатия). Наиболее популярные растровые форматы:

Bit MaP image (BMP) – универсальный формат растровых графических файлов, используется в операционной системе Windows. Этот формат поддерживается многими графическими редакторами, в том числе редактором Paint. Рекомендуется для хранения и обмена данными с другими приложениями.

Tagged Image File Format (TIFF) – формат растровых графических файлов, поддерживается всеми основными графическими редакторами и компьютерными платформами. Включает в себя алгоритм сжатия без потерь информации. Используется для обмена документами между различными программами. Рекомендуется для использования при работе с издательскими системами.

Graphics Interchange Format (GIF) – формат растровых графических файлов, поддерживается приложениями для различных операционных систем. Включает алгоритм сжатия без потерь информации, позволяющий уменьшить объем файла в несколько раз. Рекомендуется для хранения изображений, создаваемых программным путем (диаграмм, графиков и так далее) и рисунков (типа аппликации) с ограниченным количеством цветов (до 256). Используется для размещения графических изображений на Web-страницах в Интернете.

Portable Network Graphic (PNG) – формат растровых графических файлов, аналогичный формату GIF. Рекомендуется для размещения графических изображений на Web-страницах в Интернете.

Joint Photographic Expert Group (JPEG) – формат растровых графических файлов, который реализует эффективный алгоритм сжатия (метод JPEG) для отсканированных фотографий и иллюстраций. Алгоритм сжатия позволяет уменьшить объем файла в

десятки раз, однако приводит к необратимой потере части информации. Поддерживается приложениями для различных операционных систем. Используется для размещения графических изображений на Web-страницах в Интернете.

Двоичное кодирование звука

Использование компьютера для обработки звука началось позднее, нежели чисел, текстов и графики.

Звук – волна с непрерывно изменяющейся амплитудой и частотой. Чем больше амплитуда, тем он громче для человека, чем больше частота, тем выше тон.

Звуковые сигналы в окружающем нас мире необычайно разнообразны. Сложные непрерывные сигналы можно с достаточной точностью представлять в виде суммы некоторого числа простейших синусоидальных колебаний.

Причем каждое слагаемое, то есть каждая синусоида, может быть точно задана некоторым набором числовых параметров – амплитуды, фазы и частоты, которые можно рассматривать как код звука в некоторый момент времени.

В процессе кодирования звукового сигнала производится его временная дискретизация– непрерывная волна разбивается на отдельные маленькие временные участки и для каждого такого участка устанавливается определенная величина амплитуды.

Таким образом непрерывная зависимость амплитуды сигнала от времени заменяется на дискретную последовательность уровней громкости.

Каждому уровню громкости присваивается его код. Чем большее количество уровней громкости будет выделено в процессе кодирования, тем большее количество информации будет нести значение каждого уровня и тем более качественным будет звучание.

Качество двоичного кодирования звука определяется глубиной кодирования и частотой дискретизации.

Частота дискретизации – количество измерений уровня сигнала в единицу времени.

Количество уровней громкости определяет глубину кодирования. Современные звуковые карты обеспечивают 16-битную глубину кодирования звука. При этом количество уровней громкости равно $N = 216 = 65536$.

Представление видеоинформации

В последнее время компьютер все чаще используется для работы с видеоинформацией. Простейшей такой работой является просмотр кинофильмов и видеоклипов. Следует четко представлять, что обработка видеоинформации требует очень высокого быстродействия компьютерной системы.

Что представляет собой фильм с точки зрения информатики? Прежде всего, это сочетание звуковой и графической информации. Кроме того, для создания на экране эффекта движения используется дискретная по своей сути технология быстрой смены статических картинок. Исследования показали, что если за одну секунду сменяется более 10-12 кадров, то человеческий глаз воспринимает изменения на них как непрерывные.

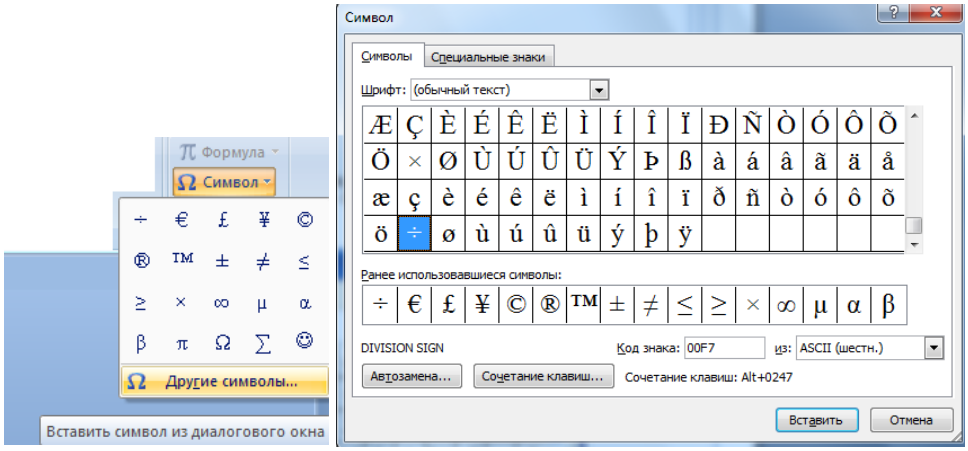
Казалось бы, если проблемы кодирования статической графики и звука решены, то сохранить видеоизображение уже не составит труда. Но это только на первый взгляд, поскольку, как показывает разобранный выше пример, при использовании традиционных методов сохранения информации электронная версия фильма получится слишком большой. Достаточно очевидное усовершенствование состоит в том, чтобы первый кадр запомнить целиком (в литературе его принято называть ключевым), а в следующих сохранять лишь отличия от начального кадра (разностные кадры).

Существует множество различных форматов представления видеоданных.

2. Пример выполнения заданий

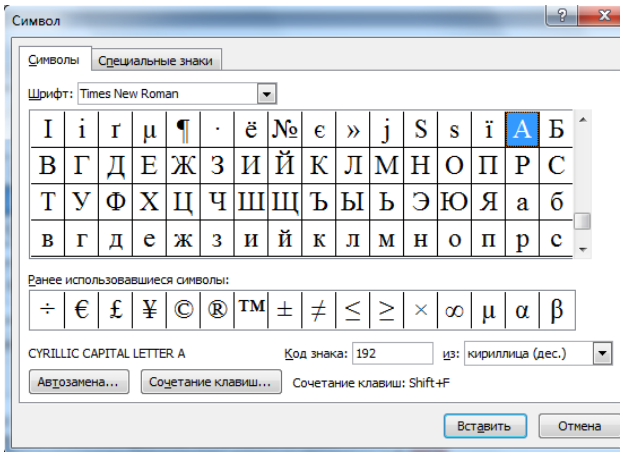
Используя таблицу символов, записать последовательность десятичных и шестнадцатеричных числовых кодов в кодировке Windows для:

Пример: Таблица символов отображается в редакторе MSWord с помощью команды: вкладка ВставкаСимволДругие символы.



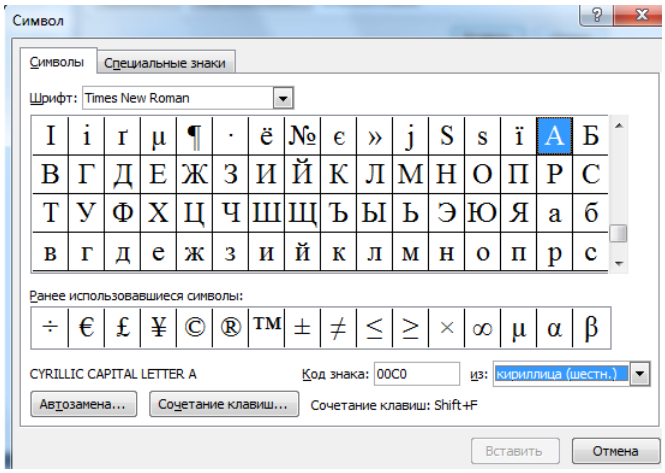
1 вариант: В поле Шрифт выбираете Times New Roman, в поле **из** выбираете кириллица (дес.):

Например, для буквы «А» (русской заглавной) код знака – 192.



2 вариант: В поле Шрифт выбираете Times New Roman, в поле из выбираете кириллица (шестн.):

Например, для буквы «А» (русской заглавной) код знака – 00C0.



Практическая часть:

Задание №1. Выписать коды заглавных (**больших**) русских букв в десятичной кодировке

А	Б	В	Г	Д	Е	Ж	З	И	К
Л	М	Н	О	П	Р	С	Т	У	Ф
Х	Ц	Ч	Ш	Щ	Ъ, Ь	Ы	Э	Ю	Я

Задание №2. Выписать коды строчных (**маленьких**) русских букв в шестнадцатеричной кодировке

а	б	в	г	д	е	ж	з	и	к
л	м	н	о	п	р	с	т	у	ф
х	ц	ч	ш	щ	ъ, ь	ы	э	ю	я

Задание №3. Записать в обеих кодировках текст «ФГБОУ ВО ИРНИТУ машиностроительный колледж»

-десятичная

Ф	Г	Б	О	У		В	О		И
Р	Н	И	Т	У		м	а	ш	и
н	о	с	т	р	о	и	т	е	л
ь	н	ы	й		к	о	л	л	е
д	ж								

- шестнадцатеричная

Ф	Г	Б	О	У		В	О		И
Р	Н	И	Т	У		м	а	ш	и
н	о	с	т	р	о	и	т	е	л
ь	н	ы	й		к	о	л	л	е
д	ж								

Задание №4. Записать в обеих кодировках название своей специальности:

-десятичная

- шестнадцатеричная

Контрольные вопросы:

1.Чем отличается непрерывный сигнал от дискретного?

2.Что такое частота дискретизации и на что она влияет?

3. В чем суть FM-метода кодирования звука?

4. В чем суть Wave-Table-метода кодирования звука?

5. Какие звуковые форматы вы знаете?

6.Какие этапы кодирования видеоинформации вам известны?

Какие форматы видео файлов Вы знаете?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 2

Тема:Работа с операционной системой, классификация программ, характеристика устройств компьютера.

Количество часов на выполнение: 2 ч.

Цель работы: Ознакомиться с основными функциями операционной системы.

– Научиться управлять файлами и папками.

– Ознакомиться с классификацией программного обеспечения.

– Изучить основные характеристики устройств компьютера.

Оборудование: Персональный компьютер

Методика выполнения задания:

Теоретическая часть

1. Операционная система (ОС)

Определение: это системное программное обеспечение, которое управляет аппаратными средствами компьютера и обеспечивает взаимодействие пользователя с системой.

Основные функции ОС:

Управление файлами и папками

Обеспечение работы программ и устройств

Предоставление пользовательского интерфейса (графический или командный)

Защита данных и управление безопасностью

Примеры ОС: Windows, macOS, Linux.

2. Классификация программного обеспечения

Операционные системы	Управляют аппаратными средствами	Windows 10, macOS, Linux
Прикладные программы	Используются для решения конкретных задач	Microsoft Word, Excel, Chrome, Photoshop
Системные утилиты	Обеспечивают обслуживание и оптимизацию системы	Антивирусы, архиваторы, дисковые утилиты
Игры	Развлекательные программы	FIFA, Minecraft, Candy Crush

Практическая часть

Задание 1. Работа с операционной системой

1. Создайте папку «Практика_Инфо» в вашей папке:
- Щёлкните правой кнопкой мыши по пустой области.
- Выберите «Создать» → «Папку».
- Назовите папку «Практика_Инфо».

2. Создайте текстовый файл внутри папки:
- Откройте программу для работы с текстом.
- Запишите в файл сведения о своих устройствах:

– Модель вашего компьютера или ноутбука (можно посмотреть в свойствах системы).

– Основные характеристики устройства: процессор, объем оперативной памяти, тип накопителя.

– Версию операционной системы (например, Windows 10, Windows 11).
- Сохраните файл в папку «Практика_Инфо» с именем «Информация о ПК».

3. Сделайте копию файла:
- Выделите файл, нажмите «Копировать» (или Ctrl+C).
- Вставьте копию в другую папку или на рабочий стол (Ctrl+V).

4. Удалите оригинальный файл из папки «Практика_Инфо».

Задание 2. Классификация программ

2.1. Составьте короткое описание следующих видов программ:

a) Операционная система (например, Windows, macOS, Linux)

b) Прикладные программы (например, Microsoft Word, Google Chrome)

c) Системные утилиты (например, антивирусы, редакторы дисков)

d) Игры

Вид программного обеспечения	Описание	Примеры программ

Задание 3. Характеристика устройств компьютера

3.1. Подготовьте сведения о характеристиках вашего устройства.

Процессор (модель, частота)

Оперативная память (объем, тип)

Жесткий диск/SSD (объем, тип)

Видеокарта (модель)

Монитор (разрешение, диагональ)

Клавиатура и мышь (тип, наличие дополнительных функций)

3.2. Запишите их в таблицу.

3.3. Обоснуйте, почему эти устройства важны для работы компьютера.

Название устройства	Характеристика	Значение или описание	Важность устройства

3.4. Используя сведения о характеристиках, ответьте на вопрос:

1. Какие устройства являются основными для корректной работы компьютера?

Контрольные вопросы:

Какие функции выполняет операционная система?

Чем отличаются системные и прикладные программы?

Назовите три основных компонента аппаратного обеспечения компьютера.

Почему важно знать характеристики устройств компьютера?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 3

Тема:Перевод чисел в разные системы счисления

Количество часов на выполнение: 2

Цель работы: Научиться переводить числа в разные системы счисления. Научиться выполнять арифметические операции в различных системах счисления; выполнять проверку с помощью Wise Calculator.

Оборудование: Персональный компьютер, Wise Calculator.

Методика выполнения задания
Теоретическая часть

1. Представление чисел в римской системе счисления.

Алфавит римской системы счисления: I(1), V(5), X(10), L(50), C(100), D(500), M(1000).

Величина в римской системе счисления определяется как сумма или разность цифр в числе. Если меньшая цифра стоит слева от большей, то она вычитается, если справа – прибавляется.

Например, запись десятичного числа 2987 в римской системе счисления будет выглядеть следующим образом: MMCMXXCVII= 2000+(1000-100)+(100-10-10)+(5+1+1).

2. Позиционные системы счисления

Система счисления	Основание	Алфавит цифр
Десятичная	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Двоичная	2	0,1
Восьмеричная	8	0, 1, 2, 3, 4, 5, 6, 7
Шестандцатеричная	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15)

3. Перевод чисел из 2-ой, 8-ой, 16-ой в десятичную систему счисления.

Для перевода числа из 2-ой, 8-ой, 16-ой в десятичную систему счисления необходимо его представить в развернутой форме и вычислить. Развернутая форма числа в двоичной системе счисления: $\boxed{A_2 = a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + ... + a_0 \cdot 2^0 + a_{-1} \cdot 2^{-1} + ... + a_{-m} \cdot 2^{-m}}$, где a_i - цифры двоичного числа (0 или 1).

Например, $11{,}01_2 = 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 2 + 1 + \frac{1}{2} + \frac{1}{2^2} = 3\frac{3}{4} = 3{,}75$

4. Алгоритм перевода целых чисел из 10-ой в 2-ую, 8-ую, 16-ую системы счисления.

1). Последовательно выполнять деление исходного целого десятичного числа и получаемых целых частных на основание системы (на 2, либо на 8, либо на 16) до тех пор пока не получится частное, меньшее делителя.

2). Записать полученные *остатки* в обратной последовательности.

Например, $19 \to A_2$

19:2 = 9 (остаток 1)

9:2 = 4 (остаток 1)

4:2=2 (остаток 0)

2:2=1 (остаток 0)

Ответ: $19_{10} = 10011_2$

5. Алгоритм перевода дробных чисел 10-ой в 2-ую, 8-ую, 16-ую системы счисления.

1). Последовательно выполнять умножение исходной десятичной дроби и получаемых дробных часть произведений на основание системы (на 2, или на 8, или на 16) до тех пор, пока не получится нулевая дробная часть или не будет достигнута требуемая точность вычислений.

2). Записать полученные целые части произведения в прямой последовательности.

В случае, если целая часть десятичной дроби отлична от нуля, то она переводится по выше указанному правилу (делением на основание той системы счисления, в которую переводим число – п.4). Дробная часть отдельно по данному правилу (смотри п. 5)

Например, $0{,}75_{10} \to A_2$

$0{,}75 \cdot 2 = 1{,}5$

$0{,}5 \cdot 2 = 1{,}00$ дробная часть получилась равной нулю, следовательно, процесс деления

прекращаем. В итоге получили $0{,}75_{10} = 0{,}11_2$

6. <u>Перевод чисел из двоичной системы счисления в 8-ую, 16-ую и обратно.</u>

Для перевода двоичного числа в 8-ую систему счисления его нужно разбить на группы по три цифры от запятой в разные стороны (для целой части группа формируется начиная справа налево и дополняется нулями в случае, если в последней (левой) группе окажется цифр меньше, чем три, то группа дополняется слева нулями; для дробной части группа формируется, начиная слева направо по три цифры в группе. Нули, в случае необходимости, добавляются справа от дробной части).

В дальнейшем задание «Перевести число из одной системы счисления в другую» будет обозначено следующим образом: $A_p \to A_q$***, где p и q могут быть равны 2, 10, 8, 16***

1. ***Арифметические операции в позиционных системах счисления***.

Сложение

Рассмотрим основные арифметические операции: сложение, вычитание, умножение и деление. Правила выполнения этих операций в десятичной системе хорошо известны — это сложение, вычитание, умножение столбиком и деление углом. Эти правила применимы и ко всем другим позиционным системам счисления.

Сложение в двоичной системе

+	0	1
0	0	1
1	1	10

Сложение в восьмеричной системе

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

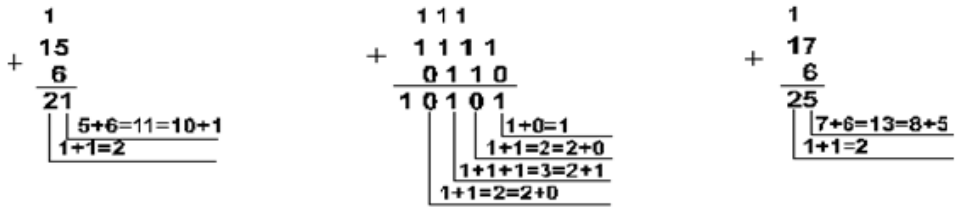
Сложение в шестнадцатеричной системе

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

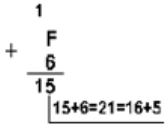
При сложении цифры суммируются по разрядам, и если при этом возникает избыток, то он переносится влево.

Пример 1. Сложим числа 15 и 6 в различных системах счисления.

Десятичная: $15_{10}+6_{10}$ **Двоичная:** 1111_2+110_2 **Восьмеричная:** 17_8+6_8



Шестнадцатеричная: F16+616



Вычитание

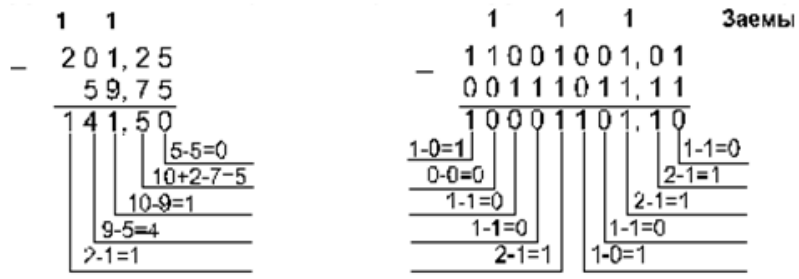
При вычитании цифры вычитаются по разрядам, и если при этом возникает недостаток, то происходит заем в старших разрядах.

Пример. Вычтем единицу из чисел 10_2, 10_8 и 10_{16}

Двоичная: 10_2-1_2 **Восьмеричная:** 10_8-1_8 **Шестнадцатеричная:** $10_{16}-1_{16}$

Пример. Вычтем число 59,75 из числа 201,25.

Десятичная: $201,25_{10} - 59,75_{10}$ **Двоичная:** $11001001,01_2 - 111011,11_2$



Восьмеричная: $311,2_8 - 73,6_8$ **Шестнадцатеричная:** $C9,4_{16} - 3B,C_{16}$

Ответ: $201,25_{10} - 59,75_{10} = 141,5_{10} = 10001101,1_2 = 215,4_8 = 8D,8_{16}$.

Умножение

Выполняя умножение многозначных чисел в различных позиционных системах счисления, можно использовать обычный алгоритм перемножения чисел в столбик, но при этом результаты перемножения и сложения однозначных чисел необходимо заимствовать из соответствующих рассматриваемой системе таблиц умножения и сложения.

Умножение в двоичной системе

*	0	1
0	0	0
1	0	1

Умножение в восьмеричной системе

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Ввиду чрезвычайной простоты таблицы умножения в двоичной системе, умножение сводится лишь к сдвигам множимого и сложениям.

Пример. Перемножим числа 5 и 6.

Десятичная: $5_{10} \cdot 6_{10}$ **Двоичная:** $101_2 \cdot 110_2$ **Восьмеричная:** $5_8 \cdot 6_8$

```
  5          101          5
× 6        × 110        × 6
 30         101          36
           101
          11110
```

Ответ: $5*6 = 30_{10} = 11110_2 = 36_8$.

Проверка. Преобразуем полученные произведения к десятичному виду:

$11110_2 = 2^4 + 2^3 + 2^2 + 2^1 = 30$;

$36_8 = 3 \cdot 8^1 + 6 \cdot 8^0 = 30$.

Деление

Деление в любой позиционной системе счисления производится по тем же правилам, как и деление углом в десятичной системе. В двоичной системе деление выполняется особенно просто, ведь очередная цифра частного может быть только нулем или единицей.

Пример 1. Разделим число 30 на число

Десятичная: $30_{10} : 6_{10}$ **Двоичная:** $11110_2 : 110_2$ **Восьмеричная:** $36_8 : 6_8$

```
 30 | 6        11110 | 110        36 | 6
 30 | 5          110 | 101        36 | 5
  0               110                0
                  110
                    0
```

Ответ: $30 : 6 = 5_{10} = 101_2 = 5_8$.

Пример 2. Разделим число 5865 на число 115.

Десятичная: $5865_{10} : 115_{10}$ **Двоичная:** $1011011101001_2 : 1110011_2$

```
 5865 | 115        1011011101001 | 1110011
  575 |  51           1110011    |  110011
  115                 1000100
  115                 1110011
    0                  10101100
                        1110011
                        1110011
                              0
```

Восьмеричная: $13351_8 : 163_8$

```
 13351 | 163
  1262 |  63
   531
   531
     0
```

Ответ: $5865 : 115 = 51_{10} = 110011_2 = 63_8$.

Проверка. Преобразуем полученные частные к десятичному виду:

$110011_2 = 2^5 + 2^4 + 2^1 + 2^0 = 51$; $63_8 = 6*8^1 + 3*8^0 = 51$.

Практическая часть

Задание 1. Представьте числа в римской системе счисления

1. 3973.
2. 4876.
3. 2742.
4. 5328.
5. 4999

Задание 2. Переведите значения из двоичной системы счисления в десятеричную:

1. $1101,01_2 \rightarrow A_{10}$
2. $101,1101_2 \rightarrow A_{10}$
3. $11101,01_2 \rightarrow A_{10}$
4. $1011,011_2 \rightarrow A_{10}$
5. $1111,01_2 \rightarrow A_{10}$

Задание 3. Переведите значения из восьмеричной системы счисления в десятеричную:

1. $74_8 \rightarrow A_{10}$
2. $65_8 \rightarrow A_{10}$
3. $57_8 \rightarrow A_{10}$
4. $61_8 \rightarrow A_{10}$
5. $74_8 \rightarrow A_{10}$

Задание 4. Переведите значения из десятичной системы счисления в двоичную, десятеричную, восьмеричную и шестнадцатеричную систему счисления:

1. $36_{10} \rightarrow A_2, A_8, A_{16}$
2. $38_{10} \rightarrow A_2, A_8, A_{16}$
3. $96_{10} \rightarrow A_2, A_8, A_{16}$
4. $78_{10} \rightarrow A_2, A_8, A_{16}$
5. $54_{10} \rightarrow A_2, A_8, A_{16}$

Задание 5. Проведите сложение, вычитание, умножение и деление двоичных чисел. Представьте результаты в десятичной системе счисления

1. 1010_2 и 10_2
2. 1110_2 и 1001_2
3. 10102 и 1012
4. 1111 2 и 112
5. 1111 2 и 1102

Контрольные вопросы:

1. Система счисления – это…
2. Чем отличаются позиционные системы счисления от непозиционных?
3. Может ли в качестве цифры использоваться символ буквы? Если да, то в какой системе счисления?
4. Что такое бит, байт?
5. Почему удобно информацию представлять в двоичных кодах?
6. Что такое двоичная триада и двоичная тетрада?
7. Объясните принцип разложения десятичного числа в степенной ряд.
8. Что такое основание системы счисления?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:[1]

Практическая работа № 4

Тема: Построение и анализ таблиц истинности в табличном процессоре. Построение логических схем и решение логических задач.

Количество часов на выполнение: 2.

Цель работы: Научиться создавать таблицы истинности для логических выражений. Освоить работу с табличным процессором Познакомиться с основами построения логических схем. Отработать навыки решения логических задач.

Оборудование: Персональный компьютер, Excel

Методика выполнения задания
Теоретический материал

1. Основы логики и таблица истинности

Логические переменные: принимают значения 0 или 1 (ложь или истина).

Логические операции:

Операция	**Обозначение**	**Описание**	**Истинность**
НЕ (отрицание)	¬A или !A	Инвертирует значение	¬0 = 1, ¬1 = 0
И (конъюнкция)	A ∧ B	Истина, если A=1 и B=1	1 ∧ 1 = 1, иначе 0
ИЛИ (дизъюнкция)	A ∨ B	Истина, если хотя бы один равен 1	0 ∨ 1 = 1, 1 ∨ 0 = 1, 0 ∨ 0 = 0
Исключающее ИЛИ (XOR)	A ⊕ B	Истина, если значения разные	1 ⊕ 0 = 1, 0 ⊕ 1 = 1, 1 ⊕ 1 = 0, 0 ⊕ 0 = 0

Таблица истинности — таблица, в которой перечислены все возможные комбинации значений переменных и соответствующие значения логического выражения

2. Построение таблиц истинности в табличном процессоре

В табличном процессоре легко можно перечислить все варианты переменных, задать формулы для логических операций с помощью встроенных функций (например, И (AND), ИЛИ (OR), НЕ (NOT)).

Таблицы позволяют быстро анализировать поведение логического выражения при разных входных данных.

3. Логические схемы

Логическая схема — графическое представление логического выражения, которое состоит из логических элементов: элементов «И», «ИЛИ», «НЕ» и др.

Позволяет визуализировать и упростить сложные логические формулы.

Методические указания по выполнению работы

Инструменты

Табличный процессор (Microsoft Excel, LibreOffice Calc, Google Sheets).

Бумага и ручка для набросков логических схем (можно использовать онлайн-сервисы для рисования схем).

Практическая часть:

Задание 1. Построение таблицы истинности

1. Выберите две логические переменные: A и B.
2. Создайте в табличном процессоре таблицу, в которой будут все комбинации A и B:

A	**B**	**…**
0	0	
0	1	
1	0	
1	1	

Задание 2. Анализ таблиц истинности

1. Проанализируйте полученные таблицы: при каких значениях переменных выражение истинно.
2. Запишите выводы для каждого выражения.

Задание 3. Построение логической схемы

Выберите одно из логических выражений из задания 1 (например, A AND (NOT B)).

Нарисуйте логическую схему, состоящую из следующих элементов:

Входы A и B.

Элемент НЕ для B.

Элемент И для результата A и ¬B.

Выход результата.

Обозначьте все элементы на схеме.

Задание 4. Решение логической задачи

Рассмотрите задачу:

"Дверь открывается, если включен ключ и либо кнопка нажата, либо датчик открыт."

Обозначьте переменные:

К — включен ключ (1 — да, 0 — нет),

Кн — кнопка нажата (1 — да, 0 — нет),

Д — датчик открыт (1 — да, 0 — нет).

Запишите логическое выражение, описывающее условие:

О = К AND (Кн OR Д)

Постройте таблицу истинности для этого выражения.

Определите при каких условиях дверь открывается.

Контрольные вопросы:

Что такое таблица истинности и какую информацию она дает?
Какие логические операции вы знаете и как они работают?
Как построить таблицу истинности с помощью табличного процессора?
Как устроена логическая схема, и зачем она нужна?
Как решить логическую задачу с помощью таблицы истинности?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 5

Тема: Методы поиска в Интернете

Количество часов на выполнение: 2.

Цель работы: уметь осуществлять поиск необходимой информации в Интернете; защита личного информационного пространства и сохранение конфиденциальности; правила организации личной информации в сети Интернет.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретические сведения.

Методы поиска информации в Интернете.

Поиск информации в Интернете осуществляется с помощью специальных программ, обрабатывающих запросы — информационно-поисковых систем (ИПС).Существует несколько моделей, на которых основана работа поисковых систем, но исторически две модели приобрели наибольшую популярность — **это поисковые каталоги и поисковые указатели.**

Поисковые каталоги устроены по тому же принципу, что и тематические каталоги крупных библиотек. Они обычно представляют собой иерархические гипертекстовые меню с пунктами и подпунктами, определяющими тематику сайтов, адреса которых содержатся в данном каталоге, с постепенным, от уровня к уровню,

уточнением темы. Поисковые каталоги создаются вручную. Высококвалифицированные редакторы лично просматривают информационное пространство WWW, отбирают то, что по их мнению представляет общественный интерес, и заносят в каталог.

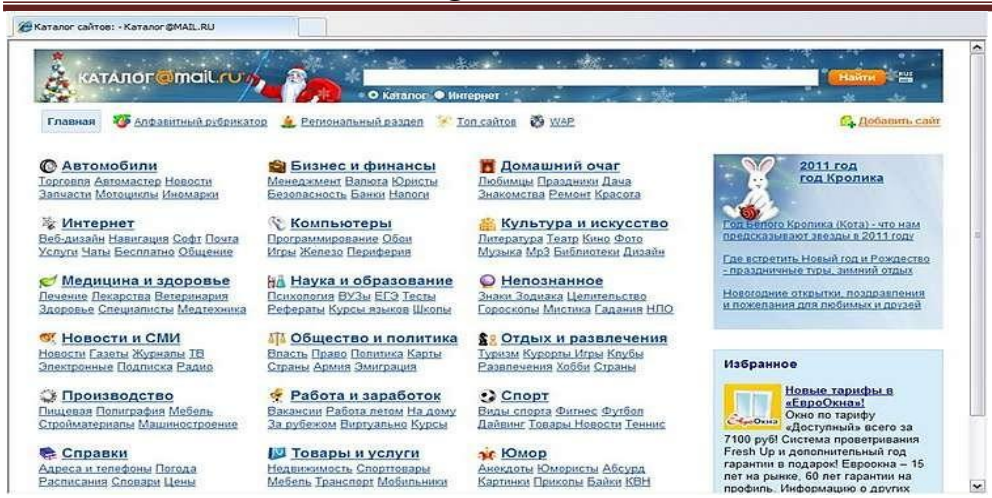
Примеры поисковых каталогов:

Атрус – www.atrus.ru.

Ау! - www.au.ru.

Каталог@mail.ru – www.list.mail.ru.

Поисковый каталог Каталог@mail.ru



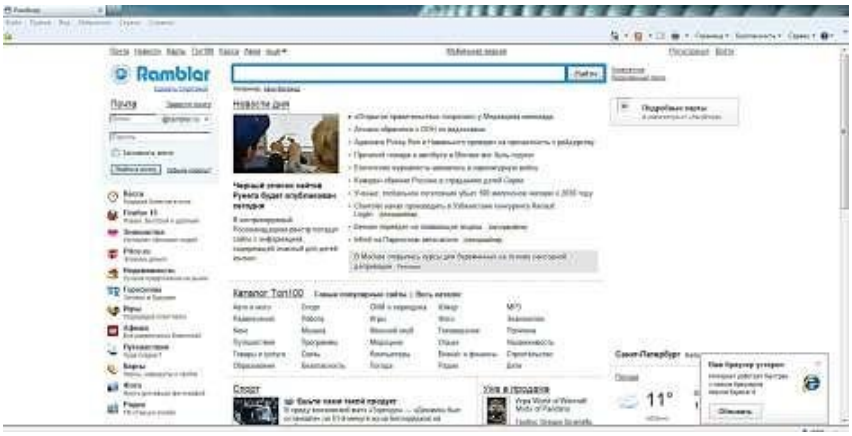
Автоматическую каталогизацию Web-ресурсов и удовлетворение запросов клиентов выполняют **поисковые указатели**. Работу поискового указателя можно условно разделить на три этапа:

- ✓ **сбор первичной базы данных**. Для сканирования информационного пространства WWW используются специальные агентские программы — черви, задача которых состоит в поиске неизвестных ресурсов и регистрация их в базе данных;
- ✓ **индексация базы данных** — первичная обработка с целью оптимизации поиска. На этапе индексации создаются специализированные документы — собственно поисковые указатели;
- ✓ **рафинирование результирующего списка**. На этом этапе создается список ссылок, который будет передан пользователю в качестве результирующего.
  Рафинирование результирующего списка заключается в *фильтрации* и р*анжировании* результатов поиска. Под *фильтрацией* понимается отсев ссылок, которые нецелесообразно выдавать пользователю (например, проверяется наличие дубликатов). *Ранжирование* заключается в создании специального порядка
  представления результирующего списка (по количеству ключевых слов, сопутствующих слов и др.).

В России наиболее крупными и популярными поисковыми указателями являются:

Яndex» (www.yandex.ru)

«Рамблер» (www.rambler.ru)



«Google» (www.google.ru)

Поиск информации - задача, которую человечество решает уже многие столетия. По мере роста объема информационных ресурсов, потенциально доступных одному человеку, были выработаны все более изощренные и совершенные поисковые средства и приемы, позволяющие найти необходимый документ. Обширные возможности для работы с большими массивами информации дают поисковые сервисы Interner.

При наличии первичных сведений по теме поиска, документы можно разыскивать поисковых системах. При этом следует различать приемы *простого,*

расширенного, контекстного и специального поиска.

Под ***простым поиском*** понимается поиск Web-ресурсов по одному или нескольким ключевым словам. Недостаток простого поиска заключается в том, что обычно он выдает слишком много документов, среди которых трудно выбрать наиболее подходящие.

При использовании ***расширенного поиска*** ключевые слова связывают между собой операторами логических отношений. Расширенный поиск применяют в тех случаях, когда приемы простого поиска дают слишком много результатов. С помощью логических отношений поисковое задание формируют так, чтобы более точно детализировать задание и ограничить область отбора, например по дате публикации или типу данных.

Контекстный поиск – это поиск по точной фразе. Он удобен для реферативного поиска информации, но доступен далеко не во всех поисковых системах. Прежде всего, чтобы обеспечивать такую возможность, система должна работать не только индексированными файлами, но и с полноценными образами Web-страниц. Эта операция достаточно медленная, и ее выполняют не все поисковые системы.

Специальный поиск применяют при розыске Web-страниц, содержащих ссылки на заданные адреса URL, содержащих заданные данные в служебных полях, например в поле заголовка и т. п.

Расширенный поиск. Кроме средства простого поиска обычно поисковые службы предоставляют средства расширенного поиска. Эти средства позволяют более точно формулировать поисковое задание, но требуют определенного опыта и работают заметно медленнее. В большинстве поисковых систем команды расширенного поиска формируются с помощью логических команд. Удобство использования логических команд в частности связано с тем, что команды простого поиска у многих поисковых систем реализованы по-разному. Каждая система стремится сделать средства простого поиска наиболее удобными, а средства расширенного поиска – наиболее стандартными. Тем не менее, для обозначения логических операторов в различных поисковых системах используются разные обозначения. Поэтому желательно перед осуществлением расширенного поиска желательно изучить синтаксис поисковых запросов выбранной поисковой системы.

Рассмотрим подробнее операторы логических отношений (логические команды).

Логическая оператор OR (ИЛИ) служит для формирования поискового запроса, если искомый текст должен содержать хотя бы один из терминов, соединенных данным оператором. Этот оператор в различных поисковых системах может обозначаться одним из следующих способов: | ; OR; ИЛИ.

Например, результат запроса «Чёрное OR море» - будет представлен списком ссылок на документы, в которых есть слово «Чёрное», или слово «море», или оба этих слова вместе.

В некоторых поисковых системах, как отмечалось выше, по умолчанию ключевые слова в запросе связаны именно этим логическим отношением.

С помощью логического оператора **AND (И)** осуществляется поиск документов, содержащих все термины, соединенные данным оператором. Этот оператор может обозначаться одним из следующих способов: **+; AND; &; И.**

Например, по запросу – «Чёрное AND море» - будут найдены документы, в которых содержатся слова «черное» и «море».

Логической оператор **NOT (НЕ)** позволяет производить поиск документов, в тексте которых отсутствуют термины, следующие за данным оператором. Этот оператор может обозначаться одним из следующих способов: **not; !; ~; НЕ.**

Например, по запросу – «Чёрное NOT море», результат - документы, в которых есть слово «Чёрное» и нет слова «море».

С помощью логических операций можно создавать достаточно сложные запросы. Запрос из нескольких слов, перемежающихся операторами, будет истолкован в

соответствии с их приоритетом. Операторы AND и NOT традиционно имеют более высокий приоритет, поэтому запрос из нескольких слов при обработке сначала группируется по операторам AND и NOT, и лишь потом по операторам OR. Например, по запросу «Чёрное AND море OR Крым» будут найдены документы, либо содержащие обязательно слова: «Чёрное» и «море», либо слово «Крым», либо все три слова.

Изменить порядок группировки можно использованием скобок. Оператор, стоящий в скобках, будет выполняться в первую очередь Использование скобок позволяет строить вложенные запросы и передавать их операторам в качестве аргументов. Так по запросу «Чёрное AND (море OR Крым)» будут найдены документы, в которых обязательно содержится слово «Чёрное» и одно из двух слов «море» или «Крым».

С помощью вложенных запросов можно значительно ограничивать область отбора, освобождая результирующий список от ненужных ссылок. Так, например, если нас интересует информация об отдыхе на юге на море, но исключительно на российском побережье, то можно попробовать использовать примерно такой запрос – «отдых AND ((Азовское OR Чёрное) AND море) NOT (Крым OR Турция OR Болгария)».

Использование круглых скобок для управления порядком исполнения задания на поиск разрешается большинством крупнейших поисковых систем.

Расширенный поиск *****

Оператор	Описание
"	Обнаруживает точные слова в кавычках или фразы
|	Найти любое из слов. Достаточно поставить между словами символ |, и вы получите страницы, где содержится хоть одно из слов запроса.
~	Исключает страницы, содержащие слово или фразу.
()	Вы можете строить сколь угодно сложные конструкции, подставляя в каждом из операторов вместо отдельного слова целые выражения. Чтобы Яндекс при этом правильно понимал вас, заключайте выражения в круглые скобки.
&	Ограничить поиск страницами, где слова запроса находятся в пределах предложения
&&	Если вам нужны документы, где присутствуют заданные слова — неважно, на каком расстоянии и в каком порядке — соедините их оператором
!	Слова с большой и маленькой буквы считаются разными формами одного слова, поэтому все равно, какой регистр использовать в запросе. Исключением является оператор точной формы. Это полезно, если искомое имя собственное совпадает с распространенным словосочетанием, например, **группа! Черный кофе**. Все слова, которые вы приводите в запросе, по умолчанию ищутся с учетом морфологии. Чтобы отключить ее, используйте оператор! перед словом (без пробела).
/	Вы можете указать максимально допустимое расстояние между двумя любыми словами запроса, поставив после первого слова символ /, сразу за которым идет число, означающее расстояние.
*	Замена части слова. Журналист*
?	Замена любого символа. Журналистик?

+	Используется для включения общих слов
"	Слова запроса, заключенного в двойные кавычки, ищутся в документах именно в том порядке и в тех формах, в которых они встретились в запросе. Таким образом, двойные кавычки можно использовать и просто для поиска слова в заданной форме (по умолчанию слова находятся во всех формах).
NOT	Оператор NOT позволяет сформировать запрос, которому отвечают документы, удовлетворяющие левой части запроса и не удовлетворяющие правой. Так, результатом поиска по запросу собака NOT кошка будут все документы, в которых есть слово "собака" и нет слова "кошка". Это особенно полезно в случаях, если искомая словоформа является одновременно формой другого слова: Женя - имя собственное, а также деепричастие. Если мы ищем именно человека по имени Женя, в запросе можно написать Женя NOT женить.
&&	Два запроса, соединенные оператором &&, образуют сложный запрос, которому удовлетворяют только те документы, которые одновременно удовлетворяют обоим этим запросам. Иными словами, по запросу собака && кошка найдутся только те документы, которые содержат и слово "собака", и слово "кошка".
||	Сложному запросу, состоящему из двух запросов, соединенных оператором ||, удовлетворяют все документы, удовлетворяющие хотя бы одному из этих двух запросов. По запросу *собака* || *кошка* найдутся документы, в которых есть хотя бы одно из двух слов - слово "собака" или слово "кошка" (или оба эти слова вместе).
()	Использование скобок позволяет строить вложенные запросы и передавать их операторам в качестве аргументов, а также перекрывать приоритеты операторов, принятые по умолчанию.

Многие поисковые системы позволяют разыскивать Web-документы по тексту, содержащемуся в заголовках. Поиск по заголовкам существенно уменьшает количество найденных ссылок, но очень точно выводит на нужные материалы. Ведь каждая Web-страница может иметь заголовок, если её автор не поленился его создать. И заголовок Web-страницы обычно точно характеризует тему материала, который содержится на ней.

Например, если вам нужна информация о дистанционном обучении, то целесообразно искать страницы, в которых это сочетание присутствует в заголовке. Таким образом, в отличие от простого запроса мы отсекаем те документы, где эти слова не являются значимыми, т. е. не определяют тему статьи.

Оператором или командой такого поиска является title, Этот оператор может обозначаться одним из следующих способов: title:, t:, title =, $title и т. п. После оператора следую ключевые слова. В некоторых поисковых системах ключевые слова следует заключать в скобки, в других они пишутся без скобок. Например, в Яndex команда поиска в заголовке записывается так: $title (очное обучение).

Поиск по сайту. С помощью поисковых систем можно осуществлять поиск информации не во всем Web пространстве, а на каком то конкретном сайте (если конечно последний проиндексирован поисковой системой). Соответствующий оператор может обозначаться следующим одним из следующих способов: url=, url:, u:, #url=". Далее следует адрес Web-узла, некоторые системы требуют заключения адреса в кавычки.

Если в запросе просто записать данный оператор с адресом какого-либо Web-узла, то будет получен список документов, проиндексированных поисковой системой на данном сайте. Но этот оператор можно комбинировать с другими, тем самым, осуществляя поиск информации по всем правилам построения запросов на данном

сайте. Например, если мы хотим найти информацию об очном обучении на сайте БелГУ, то запрос в поисковой системе Апорт будет выглядеть следующим способом: URL=http://www. bsu. ***** + “очное обучение”.

Кроме того, поисковые системы могут предлагать другие возможности специального поиска: поиск по тексту ссылок, поиск в описании документа, поиск в списке ключевых слов Web-страниц, поиск по подписям к рисункам и т. п. Следует знать, что синтаксис поисковых запросов, да и состав доступных операторов, отличается в различных поисковых указателях. Поэтому перед осуществлением поиска в той или иной поисковой системе, следует изучить страницу справки по поиску в данной системе.

В настоящее время существует множество справочных служб Интернет, помогающих пользователям найти нужную информацию. В таких службах используется обычный принцип поиска в неструктурированных документах– по ключевым словам.

Поисковая система– это комплекс программ и мощных компьютеров, способные принимать, анализировать и обслуживать запросы пользователей по поиску информации в Интернет. Поскольку современное Web- пространство необозримо, поисковые системы вынуждены создавать свои базы данных по Web- страницам. Важной задачей поисковых систем является постоянное поддержание соответствия между созданной информационнойбазой и реально существующими в Сети материалами. Для этого специальные программы (роботы) периодически обходят имеющиеся ссылки и анализируют их состояние. Данная процедура позволяет удалять исчезнувшие материалы и по добавленным на просматриваемые страницы ссылкам обнаруживать новые.

Служба World Wide Web (WWW)– это единое информационное пространство, состоящее из сотен миллионов взаимосвязанных электронных документов.

Отдельные документы, составляющие пространство Web, называют **Web-страницами**.

Группы тематически объединенных Web-страниц называют **Web-узлами**

(сайтами).

Программы для просмотра Web-страниц называют **браузерами** (обозревателями).

К средствам поисковых систем относится язык запросов.

Используя различные приёмы можно добиться желаемого результата поиска.

!– запрет перебора всех словоформ.

+– обязательное присутствие слов в найденных документах.

-– исключение слова из результатов поиска.

&– обязательное вхождение слов в одно предложение.

~– требование присутствия первого слова в предложении без присутствиявторого.

|– поиск любого из данных слов.

«»– поиск устойчивых словосочетаний.

$title– поиск информации по названиям заголовков.

$anchor–поиск информации по названию ссылок.

Поиск необходимой информации в Интернете можно осуществлять

различными способами:

1. Поиск с помощью поисковых машин по ключевому слову
2. Поиск с помощью классификаторов поисковых машин
3. Каталоги и коллекции ссылок (более общие понятия)
4. Рейтинги (самые популярные ресурсы)
5. Конференции, чаты
6. Страницы ссылок (“Links”) на тематических сайтах (редкие, специализированные вещи)
7. Несетевые способы (советы друзей, знакомых; реклама в печатных изданиях)

Поиск информации – одна из самых востребованных на практике задач, которую приходится решать любому пользователю Интернета. Существуют три основных способа поиска информации в Интернет:

1.Указание адреса страницы.

2.Передвижение по гиперссылкам.

3.Обращение к поисковой системе (поисковому серверу).

В начале поиска информации необходимо определить ее тип. Условно можно выделить 4 типа информации.

1 тип — общая (например: история Российской империи),

2 тип — менее общая (например: император Александр II),

3 тип — конкретная (например: реформы Александра II),

4 тип — более конкретная (например: отмена крепостного права).

В зависимости от типа информации определяются и пути поиска.

Информация 1 типа ищется с помощью классификаторов поисковых машин (из российских — рекомендуется Яндекс www.Yandex.ru). Если сразу сайты с требуемой информацией не находятся, то следует просматривать найденные по классификатору каталоги и страницы ссылок (“Links”), которые находятся сайтах подобной тематике. Эти сайты приводятся в классификаторе по теме и найденных каталогах.

Информация 2 типа ищется подобно поиску для 1 типа, но с преимуществом поиска по каталогам и страницам ссылок.

Информация 3 типа — по ключевым словам, которые вводятся в строку поиска поисковых машин, каталогам, страницам ссылок

Информация 4 типа — по подробным данным, которые вводятся в строку поиска. Данные находятся согласно способам поиска изложенных для 2 и 3 типов.

С развитием INTERNET появилась возможность быстрого и удобного поиска необходимой документальной информации. Теперь можно не заниматься подбором и изучением огромного количества литературы в книжных магазинах и библиотеках. Информацию можно получить, не выходя из дома или офиса. Для этого нужен только непосредственно сам компьютер, подключенный к INTERNET с установленной специальной программой – браузером, предназначенной для просмотра содержимого Web-страниц.

Благодаря разнообразию поисковых систем, специально разработанным для рядового пользователя, каждый может без труда отсечь заведомо ненужный поток информации, лишь правильно сформулировав цель поиска.

Личное информационное пространство – это понятие, которое описывает совокупность данных и информации, которые человек хранит о себе, своих интересах, своей жизни и деятельности. Это место, где накапливаются и хранятся различные сведения о личности, которые могут использоваться как в личных, так и в профессиональных целях.

Формирование личного информационного пространства – это процесс, который требует внимания и активности каждого человека. Оно начинается с осознания необходимости сохранять и контролировать свои личные данные и информацию. Чтобы сформировать свое личное информационное пространство, нужно определиться с границами и правилами доступа к своим сведениям, а также научиться управлять этим пространством.

Основной принцип формирования личного информационного пространства – это защита личных данных и сохранение конфиденциальности. Человек самостоятельно определяет, кому и какую информацию он может предоставить, а кому и какую информацию следует оставить закрытой. Для этого можно использовать различные инструменты и методы, такие как пароли, шифрование данных, контроль доступа к информации и т.д.

Вследствие этого, чтобы обезопасить себя и не искушать злоумышленников, в сети следует придерживаться следующих несложных **правил организации личной информации в сети Интернет.**

- В первую очередь необходимо правильно организовать ту информацию, которую вы указываете в ваших личных профилях на различных сайтах, социальных сетях или форумах. Старайтесь не заполнять личные анкеты на неизвестных вам сайтах, особенно это касается неизвестных социальных сетей. Как вариант – заполняйте такие анкеты по минимуму. Ведь любая социальная сеть – отличная база данных о пользователях.
- Не выставляйте свой номер мобильного или домашнего телефона на всеобщее обозрение в социальных сетях без разрешения родителей. Также это касается домашнего адреса и номера школы.
- Не высылайте никому свои фотографии, не проконсультировавшись с родителями.
- В программе Skype по видеосвязи общайтесь лишь с проверенными и знакомыми вам людьми, чтобы предотвратить попадания видео с вами в руки злоумышленников.
- Не открывайте прикрепленные к электронному письму файлы, присланные от незнакомого человека. Файлы могут содержать вирусы или другие программы, которые могут нанести вред информации, хранящейся на компьютере или программному обеспечению компьютера.
- **Никогда и ни при каких обстоятельствах не разглашайте никому информацию о своем пароле.**

Облачные сервисы. Разделение прав доступа в облачных хранилищах

Облако — термин, под которым понимают пользование веб-сервисами, запущенными на удаленных серверах, которые принадлежат и предоставлены третьими лицами, к которым можно подключиться при помощи Интернета с любого устройства - будь то персональный компьютер, рабочий ноутбук, мобильный телефон или планшет.

Облачные технологии – это возможность иметь доступ к данным, не устанавливая специальных приложений на устройстве. Все необходимое обеспечение пользователям предоставляют серверы. Простыми словами, облачное хранилище, это сервис, который предоставляет для вас определенное место в интернете для хранения ваших файлов.

Вот один примеров использования виртуального облака:

Сейчас становится неактуально держать всю свою музыкальную коллекцию на локальном жестком диске. Например, облачный сервис, как "Яндекс музыка". Это очень удобно — иметь доступ к любой музыкальной композиции онлайн и располагать возможностью создания онлайн плейлистов.

Яндекс.Диск — бесплатный облачный сервис от Яндекса, позволяющий пользователям хранить свои данные на серверах в облаке и передавать их другим пользователям в интернете. Работа построена на синхронизации данных между различными устройствами. В настоящее время регистрация пользователей доступна всем. Ранее, до запуска Яндекс. Диска, функции хранения пользовательских файлов на Яндексе выполнял сервис Яндекс.Народ.

Изначально Яндекс.Диск предоставляет около 10 Гб навсегда.

Кроме того, Яндекс.Диск может выступать в качестве службы облачного сервиса, интегрируясь в офисный пакет Microsoft Office, а недавно появилась возможность автоматической загрузки фото и видеофайлов с цифровых камер и внешних носителей информации на Яндекс. Диск. При этом пользователю предоставляются дополнительно 32 ГБ пространства на полгода.

Методы работы с Яндекс. Диск

Сервисом Яндекс.Диск можно пользоваться двумя способами:

1) Можно заходить в папку Яндекс.Диска по публичной ссылке (публичная ссылка – это ссылка на файлы или папки, предназначенные для общего доступа), отправленной вам преподавателем или другом, и пользоваться данными.

2) Можно создавать собственные ресурсы, личные или предназначенные для общего доступа в облаке, установив Яндекс.Диск на свой компьютер.

Работа с Яндекс Диском через приложение

Перейдём к вопросу об использовании диска, рассмотрим сначала вариант с использованием приложения. Вы получаете раздел на компьютере, работающий как одно целое с серверным хранилищем, естественно вы вольны выбирать её место расположения. Затем, чтобы загрузить любую интересующую вас информацию, вам потребуется всего лишь скопировать файл в этот раздел. После этого значок приложения будет показывать состояние обработки и начнет загружать файлы на сервер. Эта папка способна поддерживать все функции Windows, вы можете как перетянуть файл в неё, так и вставить скопированный заранее элемент. Выполнив такие простые действия, вы сможете использовать облачное хранилище Яндекса.

Если вы хотите обмениваться информацией просто с другом, или выкладывать файлы на ваш блог, следует нажав правой кнопкой, выбрать пункт из меню, который называется «Яндекс.Диск: Скопировать публичную ссылку». После этого перешлите или разместите полученную ссылку и перейдя по ней любой сможет загрузить данный файл.

Как уже упоминалось ранее, указанный сервис не требует какие-либо условия для продолжительного хранения информации, она вечна по умолчанию.

Аналогичным образом вы сможете и ограничить доступ, сделав данные личными.

Яндекс Диск — использование через браузер

Для того чтобы не загружать приложение, экономя таким образом системные ресурсы вы сможете получить доступ к любой информации ранее загруженной на облако через браузер. Существуют несколько отличий в использовании этих подходов:

-Используя этот метод, вы не получаете синхронизации информации с Яндекс Диском;

-Некоторые браузер накладывают ограничение на загрузку файлов с объёмом выше 2Гб, поэтому может случиться обрыв соединения;

-Предоставляется возможность пред просмотра файлов, не все типы содержимого поддерживаются, но текстовые файлы, вроде Word, изображения и подобные, можно предварительно изучить;

-В онлайн режиме существует ещё одна важная функция, а именно корзина, из которой можно достать ошибочно удалённые файлы;

-Удобный интерфейс с различными фильтрами по поиску необходимого содержимого.

Практическая часть

Задание 1. Создайте свой почтовый ящик на Яндексе (или войдите в него).

Если вы хотите иметь 10Гбайт или даже больше памяти на серверах Яндекса для хранения резервных копий информации, размещённой на вашем компьютере, делиться событиями вашей жизни, запечатлёнными на фото и видео, тогда можно воспользоваться облачным сервисом Яндекс. Диск или другими подобными сервисами.

Для этого вам потребуется Яндекс-аккаунт, а точнее электронная почта в Яндексе.

Сервисом Яндекс. Диск можно пользоваться двумя способами:

♣ Можно заходить в папку Яндекс.Диска по публичной ссылке (публичная ссылка – это ссылка на файлы или папки, предназначенные для общего доступа), отправленной вам преподавателем или другом, и пользоваться данными.

♣ Можно создавать собственные ресурсы, личные или предназначенные для общего доступа в облаке, установив Яндекс.Диск на свой компьютер.

Задание 2. Работа с Яндекс.Диском

1. Создать папку с именем ПР05, в папке поместите его в документ *отчета ПР05.docx*
2. Войдите в аккаунт на сервисе ЯндексДиск. (сделайте скриншот)
3. Создайте папку и загрузите на Яндекс.Диск любой файл из ваших файлов документов или рисунков. (сделайте скриншот и поместите его в документ *отчета ПР05.docx)*
4. Поделитесь ссылкой на загруженный файл с одногруппниками с помощью эл. Почты, (сделайте скриншот и поместите его в документ *отчета ПР05.docx*).
5. Ознакомьтесь с Яндекс сервисами, (сделайте скриншот и поместите его в документ *отчета ПР05.docx*).
6. Попробуйте специальный поиск на сервисе Яндекс, (сделайте скриншот и поместите его в документ *отчета ПР05.docx*).
7. Перейдите в сервис Яндекс.Новости (узнайте актуальную информацию в разделе политика) (сделайте скриншот и поместите его в документ *отчета ПР05.docx*).
8. Сохранить документ *отчета ПР065docx.* Выйти из своего аккаунта, закрыть браузер.

Задание 3. *Найдите информацию об интересных фактах в Интернете.*

1. Чем уникальна улица Ковентри в Волгограде?
2. Где расположен самый высокий памятник в мире, установленный реальному человеку? Укажите его размеры и фото.

3. Какой размер имеет самый длинный дом в Европе? Укажите его адреса и фото
4. Когда Волгоград переименовывается в "город-герой Сталинград" (точные даты)
5. Чему равна протяженность самой длинной улицы России, которая не имеет официального статуса улицы? Какое название она носит?
6. Найдите фотографию Билла Гейтса.
7. Статус и состояние Билл Гейтс заработал благодаря делу всей жизни - созданию программного обеспечения для персональных компьютеров. Гейтс и его друг Пол Аллен (он и сейчас работает в Microsoft) образовали малое предприятие Micro-Soft (впоследствии дефис в названии компании отпал) для того, чтобы торговать собственноручно разработанной версией языка программирования BASIC для полулюбительского бытового вычислительного устройства Altair 8800. Сколько лет было Биллу Гейтсу, когда появилась компания Microsoft?
8. На фотографии 1978 года они стоят в три ряда - одиннадцать единомышленников, среди них только две девушки. Билл Гейтс и Пол Аллен как раз рядом с девушками. Молодые лица - и такое обилие бород, длинных волос, очков. Сразу видно, на снимке не спортсмены, и не актеры, и не клерки. У всех в глазах некая отстраненность - вот они, первые «чокнутые компьютерщики», или geeks, как их называют в Америке. Найдите эту фотографию.
9. Билл Гейтс написал две книги в 1995 и 1999 годах. В одной из них он изложил свои взгляды на то, в каком направлении движется общество в связи с развитием информационных технологий, а в другой показал, как информационные технологии могут решать бизнес-задачи в совершенно новом ключе. Как назывались эти книги?
10. Доходы от продаж обеих своих книг г-н Гейтс перечисляет в благотворительный фонд. На что направлена деятельность благотворительного фонда?
11. Чем, помимо Microsoft, ещё занимается Билл Гейтс?

Задание 4. Особенности поиска по группе слов.
Заполните таблицу, используя поисковую систему Яндекс.

Структура запроса	Количество найденных страниц	Электронный адрес первой найденной ссылки
Скоростной! трамвай!		
Скоростной + трамвай		
Скоростной - трамвай		
«Скоростной трамвай»		
Сталинградская | битва		
Сталинградская &битва		
$title (река Волга)		
$anchor (река Волга)		

Контрольные вопросы
1. Что такое Интернет?
2. Какие протоколы используются в сети Интернет?

3. Какие программы просмотра WWW(браузеры) вы можете назвать?

4. Какие средства поиска существуют в Интернет?

5. Какие средства общения предлагает Интернет?

6. Как выполнить поиск информационного объекта в текстовом процессоре MS Word?

7. Как выполнить поиск информационного объекта в файловых структурах Windows?

8. Перечислите известные вам поисковые машины.

9. Для чего в некоторых поисковых системах используется расширенный поиск?

10. Каким логическим оператором связаны ключевые слова в простом запросе в рассмотренных поисковых машинах?

11. Как в поисковой машине Яndex осуществить поиск точной формы слов?

12. Какие области поиска можно определить в Яndex?

13. Как в Яndex указать расстояние между ключевыми словами?

14. Дайте определение облачные технологии.

15. Дайте определение Яндекс.Диск.

16. Назовите методы работы с Яндекс.Диск.

17. Назовите какие еще облачные хранилища вам знакомы.

18. Для чего необходимы облачные хранилища?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 6

Тема: Организация личного информационного пространства. Разделение прав доступа в облачных хранилищах. Облачные хранилища данных.

Количество часов на выполнение: 2

Цель работы: Ознакомиться с концепцией организации личного информационного пространства.Изучить принцип разделения прав доступа в облачных хранилищах.

На практике освоить работу с облачными хранилищами данных (Google Drive, OneDrive, Dropbox и др.). Научиться создавать структуру папок, управлять правами доступа и делиться файлами.

Оборудование: Персональный компьютер, облачное хранилище

Методика выполнения задания
Теоретические сведения.

1. Что такое личное информационное пространство

Личное информационное пространство — это систематизированное место для хранения, организации, обработки и доступа к личной информации и файлам, которое позволяет удобно и безопасно управлять данными. Обычно реализуется через облачные хранилища, локальные папки и приложения.

2. Облачные хранилища данных

Облачные хранилища (cloud storage) — это сервисы, позволяющие хранить, синхронизировать и совместно использовать файлы через интернет. Основные преимущества:

Доступ из любого устройства с интернетом.

Возможность делиться файлами и папками с другими пользователями.

Центральное хранение и автоматическое резервное копирование.

Возможность организации структуры хранения.

Примеры облачных хранилищ:

Google Drive

Microsoft OneDrive

Dropbox

Yandex Disk

3. Разделение прав доступа

Облачные сервисы позволяют управлять правами доступа к файлам или папкам:

Только просмотр — пользователь может видеть файлы, но не редактировать.

Редактирование — пользователь может изменять файлы.

Комментирование — пользователь может оставлять комментарии.

Удаление — пользователь может удалять файлы (часто ограничено правами).

Настройки прав доступа могут делать ссылки публичными или ограниченными (только выбранные пользователи или группы).

Методические указания

Инструменты

Облачное хранилище (например, Google Drive или другой сервис).

Компьютер или мобильное устройство для доступа к облаку.

Интернет-соединение.

Практическая часть

Задание 1. Создание структуры личного информационного пространства

Зарегистрируйтесь или войдите в выбранное облачное хранилище.

Создайте папки для организации:

Документы

Фото

Образование

Работы/учебные материалы

В каждой папке создайте подпапки по необходимости (например, в папке «Образование» — «Учебники», «Заметки»).

Задание 2. Загрузка и организация файлов

Загрузите в каждую папку несколько файлов: документы, фотографии, учебные материалы.

Ориентируйтесь на их тематическую структуру и аккуратность.

Задание 3. Настройка прав доступа

Создайте ссылку для общего доступа к одной из папок.

Измените настройки доступа:

Сделайте её «Общедоступной» с правом «Только просмотр».

Создайте отдельную ссылку с правом «Редактирование» для выбранного друга или одноклассника.

Попробуйте открыть эти ссылки с другого устройства или аккаунта и проверьте права.

Задание 4. Совместное использование и контроль доступа

Пригласите одногруппника на доступ к папке по электронной почте.

Настройте права так, чтобы он мог редактировать файлы.

Обсудите, зачем важно управлять правами доступа и как это помогает сохранять безопасность информации.

Задание 5.

Создайте собственное «личное информационное пространство» в выбранном облачном сервисе, организуйте папки и файлы.

Настройте уровни доступа, делитесь файлами и папками с друзьями или однокурсниками, объясните, почему важно правильно управлять правами доступа.

Контрольные вопросы:

1. Чем отличается локальное хранение данных от облачного?
2. Какие преимущества дает организация личного информационного пространства?
3. Какие права доступа можно устанавливать в облачных хранилищах?
4. Почему важно ограничивать доступ к некоторым файлам или папкам?
5. Как правильно делиться файлами и управлять правами доступа?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 7

Тема: Антивирусные программы. Программы-архиваторы

Количество часов на выполнение: 2.

Цель работы: Изучить технологию тестирования компьютера на наличие вируса и профилактические меры. Познакомиться со способами лечения зараженных объектов.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Компьютерный вирус – это специально написанная, небольшая по размерам программа (т.е. некоторая совокупность выполняемого кода), которая может “ приписывать” себя к другим программам (“заражать” их), создавать свои копии и внедрять их в файлы, системные области компьютера и т.д., а также выполнять различные нежелательные действия на компьютере.

Программа, внутри которой находится вирус, называется “зараженной” Когда такая программа начинает работу, то сначала управление получает вирус. Вирус находит и заражает другие программы, а также выполняет какие-нибудь вредные действия (портит файлы или таблицу размещения файлов на диске, “засоряет” оперативную память и т.д.).

Классификация вирусов.

По среде обитания	сетевые	распространяются по компьютерной сети
	файловые	внедряются в выполняемые файлы
	загрузочные	внедряются в загрузочный сектор диска (Boot-сектор)
	файлово-загрузочые	внедряются в выполняемые файлы и в загрузочный сектор диска
	системные	проникают в системные модули и драйверы периферийных устройств, поражают программы-интерпретаторы
По способу заражения	резидентные	находятся в памяти, активны до выключения компьютера
	нерезидентные	не заражают память, являются активными ограниченное время
По деструктивным возможностям (по способам воздействия)	безвредные	практически не влияют на работу; уменьшают свободную память на диске в результате своего распространения
	неопасные	уменьшают свободную память; создают звуковые, графические и прочие эффекты
	опасные	могут привести к серьёзным сбоям в работе
	очень опасные	могут привести к потере программ или системных данных
По особенностям алгоритма вируса	вирусы-«спутники»	вирусы, не изменяющие файлы, создают для EXE-файлов файлы-спутники с расширением COM
	простейшие вирусы	паразитические программы, которые изменяют содержимое файлов и секторов диска и могут быть легко обнаружены

	Ретро-вирусы	обычные файловые вирусы, которые пытаются заразить антивирусные программы, уничтожая их, или делая неработоспособными
	репликаторные, вирусы-«черви»	распространяются по сети, рассылают свои копии, вычисляя сетевые адреса. Это самые распространенные в виртуальной сети вирусы. Они очень быстро «размножаются». Иногда дают своим копиям отдельные имена. Например, «install.exe».
	«паразитические»	изменяют содержимое дисковых секторов или файлов
	«студенческие»	примитив, содержат большое количество ошибок
	«стелс»-вирусы (невидимки)	это файловые вирусы, которых антивирусные программы не находят, потому что во время проверки они фальсифицируют ответ. Они перехватывают обращения DOS к пораженным файлам или секторам и подставляют вместо себя незараженные участки
	вирусы-призраки	не имеют ни одного постоянного участка кода, труднообнаруживаемы, основное тело вируса зашифровано
	макровирусы	пишутся не в машинных кодах, а на WordBasic, живут в документах Word, переписывают себя в шаблон Normal.dot
	квазивирусные, или «троянские»	это вирусы, не способные к «размножению». Троянская программа маскируется под полезную или интересную программу, выполняя во время своего функционирования ещё и разрушительную работу (например, стирает FAT-таблицу) или собирает на компьютере не подлежащую разглашению информацию. В отличие от вирусов, троянские программы не обладают свойством самовоспроизводства. Троянская программа маскируется, как правило, под коммерческий продукт. Её другое название «троянский конь».

	логические бомбы	программы, которые запускаются при определённых временных или информационных условиях для осуществления вредоносных действий (как правило, несанкционированного доступа к информации, искажения или уничтожения данных)
	мутанты	это один из видов вирусов, способных к самовоспроизведению. Однако их копия явно отличается от оригинала.

Основными путями проникновения вирусов в компьютер являются съёмные диски (гибкие и лазерные), а также компьютерные сети. Заражение жесткого диска вирусами может произойти при загрузке программы с дискеты, содержащей вирус. Такое заражение может быть и случайным, например, если дискету не вынули из дисковода A: и перезагрузили компьютер, при этом дискета может быть и не системной. Заражение дискеты происходит, даже если её просто вставили в дисковод зараженного компьютера или, например, прочитали её оглавление.

Признаки заражения

- вывод на экран непредусмотренных сообщений или изображений;
- подача непредусмотренных звуковых сигналов;
- неожиданное открытие и закрытие лотка CD-ROM-устройства;
- произвольный, без вашего участия, запуск на компьютере каких-либо программ;

Есть также косвенные признаки заражения вашего компьютера:

- частые зависания и сбои в работе компьютера;
- прекращение работы или неправильная работа ранее успешно работавших программ;
- медленная работа компьютера при запуске программ;
- невозможность загрузки операционной системы;
- исчезновение файлов и каталогов или искажение их содержимого;
- изменение размеров файлов;
- неожиданное значительное увеличение количества файлов на диске;
- существенное уменьшение размеров свободной оперативной памяти;
- частое обращение к жесткому диску (часто мигает лампочка на системном блоке);
- Microsoft Internet Explorer "зависает" или ведет себя неожиданным образом.

В 90% случаев наличие косвенных симптомов вызвано сбоем в аппаратном или программном обеспечении. Несмотря на то, что подобные симптомы с малой вероятностью свидетельствуют о заражении, при их появлении рекомендуем вам провести полную проверку вашего компьютера.

Антивирусные программы.

Для обнаружения, удаления и защиты от компьютерных вирусов разработаны специальные антивирусные программы. Различают следующие виды антивирусных программ:

- **Программы-детекторы** осуществляют поиск характерной для конкретного вируса сигнатуры в оперативной памяти и в файлах и при обнаружении выдают соответствующее сообщение. Недостатки: могут находить только те вирусы, которые известны разработчикам этой программы, поэтому быстро устаревают и требуют регулярного обновления.
- **Программы-доктора** или **фаги** не только находят зараженные вирусами файлы, но и «лечат» их, т.е. удаляют из файла тело программы-вируса, возвращая файл в исходное состояние. **Полифаги** – программы-доктора, предназначенные для поиска и уничтожения большого количества вирусов. Недостатки те же, что и у программ-детекторов.
- **Программы-ревизоры** относятся к самым надежным средствам защиты. Ревизоры запоминают исходное состояние программ, каталогов и системных областей диска тогда, когда компьютер не заражен вирусом, а затем периодически или по желанию пользователя сравнивают текущее состояние с исходным. Обнаруженные изменения выводятся на экран монитора.
- **Программы-фильтры** или **«сторожа»** представляют собой небольшие резидентные программы, предназначенные для обнаружения подозрительных действий при работе компьютера, характерных для вирусов (попытки коррекции файлов с расширением EXE или COM, изменение атрибутов файла, запись в загрузочные сектора и т.п.). При попытке какой-либо программы произвести указанные действия «сторож» посылает пользователю сообщение и предлагает запретить или разрешить соответствующее действие. Эти программы способны обнаружить вирус на самой ранней стадии его существования до размножения. Однако они не лечат файла и диски. Для уничтожения вируса требуется применить другие программы.
- **Вакцины** или **иммунизаторы** это резидентные программы, предотвращающие заражение файлов. Вакцины применяют, если отсутствуют программы-доктора, лечащие этот вирус. Вакцинация возможна только от известных вирусов. Вакцина модифицирует программу или диск таким образом, чтобы это не отражалось на их работе, а вирус будет воспринимать их зараженными и поэтому не внедрится. Имеют ограниченное применение.

Профилактика заражения компьютера вирусами.

Никакие самые надежные и разумные меры не смогут обеспечить стопроцентную защиту от компьютерных вирусов и троянских программ, но, выработав для себя ряд правил, вы существенно снизите вероятность вирусной атаки и степень возможного ущерба.

Одним из основных методов борьбы с вирусами является, как и в медицине, своевременная *профилактика*. Компьютерная профилактика состоит из небольшого количества правил, соблюдение которых значительно снижает вероятность заражения вирусом и потери каких-либо данных.

Ниже перечислены основные правила безопасности, выполнение которых

позволит вам избегать вирусных атак.

Правило № 1: *защитите ваш компьютер с помощью антивирусных программ и программ безопасной работы в интернете*. Для этого:

- Безотлагательно установите антивирусную программу.
- Ежедневно обновляйте антивирусные базы. Обновление можно проводить несколько раз в день при возникновениях вирусных эпидемий
- Задайте рекомендуемые настройки для постоянной защиты. Постоянная защита вступает в силу сразу после включения компьютера и затрудняет вирусам проникновение на компьютер.
- Задайте рекомендуемые настройки для полной проверки компьютера и запланируйте ее выполнение не реже одного раза в неделю.

Правило № 2: *будьте осторожны при записи новых данных на компьютер*:

- Проверяйте на присутствие вирусов все съемные диски (дискеты, CD-диски, флэш-карты и пр.) перед их использованием.
- Осторожно обращайтесь с почтовыми сообщениями. Не запускайте никаких файлов, пришедших по почте, если вы не уверены, что они действительно должны были прийти к вам, даже если они отправлены вашими знакомыми. В особенности не доверяйте письмам якобы от антивирусных производителей.
- Внимательно относитесь к информации, получаемой из интернета. Если с какого-либо веб-сайта вам предлагается установить новую программу, обратите внимание на наличие у нее сертификата безопасности.
- Если вы копируете из интернета или локальной сети исполняемый файл, обязательно проверьте его антивирусной программой.
- Внимательно относитесь к выбору посещаемых вами интернет-сайтов. Некоторые из сайтов заражены опасными скрипт-вирусами или интернет-червями.

Правило № 3: *внимательно относитесь к информации об эпидемиях компьютерных вирусов.*.

В большинстве случаев о начале новой эпидемии сообщается задолго до того, как она достигнет своего пика. Вероятность заражения в этом случае еще невелика, и, скачав обновленные антивирусные базы, вы сможете защитить себя от нового вируса заблаговременно.

Правило № 4: *с недоверием относитесь к вирусным мистификациям* - "страшилкам", письмам об угрозах заражения.

Правило № 5: *пользуйтесь сервисом Windows Update и регулярно устанавливайте обновления операционной системы Windows*.

Правило №6: *покупайте дистрибутивные копии программного обеспечения у официальных продавцов*.

Правило № 7: *ограничьте круг людей, допущенных к работе на вашем компьютере*.

Правило № 8: *уменьшите риск неприятных последствий возможного заражения*:

• Своевременно делайте резервное копирование данных. В случае потери данных система достаточно быстро может быть восстановлена при наличии резервных копий. Дистрибутивные диски, дискеты, флэш-карты и другие носители с программным обеспечением и ценной информацией должны храниться в надежном месте.

• Обязательно создайте системную аварийную дискету, с которой при необходимости можно будет загрузиться, используя "чистую" операционную систему.

Практическая часть

Задание 1. По поражаемым объектам компьютерные вирусы делятся на:

- Файловые вирусы
- Загрузочные вирусы
- Сценарные вирусы
- Макровирусы
- Вирусы, поражающие исходный код

Составьте таблицу с определения и примеры вирусов для каждого из этих классов.

Наименование	Определение	Пример

Задание 2. Найдите примеры антивирусных программ для каждого из этих видов.

- Программы-детекторы
- Программы-доктора или фаги и ПолифагиПрограммы-ревизоры
- Программы-фильтры или «сторожа»
- Вакцины или иммунизаторы

Задание 3. Запустите имеющуюся у вас антивирусную программу и проверьте наличие вирусов на локальном диске С: Опишите ход работы и сделайте скриншоты.

Задание 4. Создайте у себя на диске папку. Найдите на диске С: файлы с любым расширением. Скопируйте самый маленький по размеру из найденных файлов в папку (проведите сортировку по размеру). Проверьте папку с записанными файлами на наличие вирусов. Опишите ход работы и сделайте скриншоты.

Задание 5. Опишите ход работы и сделайте скриншоты.

Подготовка тестового “вирусного” файла

Для проверки работоспособности антивирусных программ их производители предусматривают возможность создания текстового файла, содержащего специальную текстовую строку, который все основные антивирусные программы распознают как “EICAR Test File”. Хотя данный файл обрабатывается антивирусными программами точно так же, как вирус, он не является вирусом в полном смысле этого слова и не вызывает заражения компьютера или каких-либо других нежелательных эффектов.

1. Запустите на исполнение стандартное приложение Windows — текстовый редактор Блокнот.

2. Откройте предоставленный файл документа eicar.doc и скопируйте содержащуюся в нем текстовую строку в пустое окно редактора Блокнот, либо введите эту строку с клавиатуры:

X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*

3. Дважды сохраните созданный текстовый файл в папке Мои документы под именем eicar.txt и eicar2.txt.

Проверка работы антивирусного сторожа

1. Откройте папку Мои документы и переименуйте только что сохраненный файл eicar.txt в eicar.com. Убедитесь, что сторож антивирусной программы обнаружил созданный нами “тестовый вирус”:

2. В появившемся окне предупреждения от антивирусного сторожа выберите кнопку Delete (“удалить вирус”), так как радиокнопка Repair (“лечить”) невозможна. Щелкните мышью на кнопке ОК для подтверждения операции.

3. Снова откройте окно папки Мои документы и убедитесь, что файл eicar.com перестал существовать.

Проверка работы антивирусного сканера

1. Создайте в папке Мои документы новую папку с произвольным именем. Переименуйте ранее сохраненный файл eicar2.txt в eicar.txt и переместите его в созданную папку (перетащив переименованный файл мышью на ее иконку).

2. Выполните проверку на вирусы содержимого созданной папки. Для этого щелкните на ее иконке правой кнопкой мыши и выберите в контекстном меню пункт Проверить с

3. Убедитесь, что антивирусный сканер запустился и начал проверку файлов. В появившемся окне убедитесь, что антивирусный сканер обнаружил содержащийся в вашей папке “вирус”:

Выберите кнопку Ignore (этот файл еще понадобится нам для дальнейшей практической работы) и щелкните мышью на кнопке ОК.

4. Просмотрите содержимое основного окна антивирусного сканера:

Ознакомьтесь с приведенной в нем информацией о количестве проверенных файлов папок и архивов (Scanned files, Scanned directories, Scanned archives) и количестве обнаруженных вирусов (Detections). Закройте окно, щелкнув мышью на кнопке Закрыть.

5. Проверьте, что файл остался в папке.

6. Удалите папку с файлом.

Контрольные вопросы:

1. Что такое компьютерный вирус?
2. На какие типы разделяют компьютерные вирусы в различных видах классификации?
3. Чем отличаются макровирусы от обычных загрузочных вирусов?
4. Каковы основные пути проникновения вирусов в компьютер?
5. По каким признакам можно судить о поражении компьютера вирусом?

6. Какие типы антивирусных программ вам известны?
7. Как проверить CD-диск или дискету на наличие вируса с помощью антивирусных программ?
8. В каком файле содержится информация о зараженных и вылеченных объектах?
9. Перечислите профилактические меры для борьбы с заражением вирусами.

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 8

Тема: Редактирование и форматирование текстовых документов создание оглавлений. Оформление рефератов. Вёрстка документов с таблицами, рисунками и математическими формулами.

Количество часов на выполнение: 2.

Цель работы: научиться редактировать и форматировать текстовые документы, научиться создавать и форматировать маркированные и нумерованные списки, вставлять и форматировать колонтитулы

Оборудование: Персональный компьютер

Методические указания к выполнению:

Теоретические сведения:

Microsoft Word — основа любого офиса и, пожалуй, самая нужная и популярная программа во всем Microsoft Office. С помощью Word вы можете не просто набрать текст, но и оформить его по своему вкусу: включить в него таблицы и графики, картинки и даже звуки и видеоизображения. Word поможет вам составить простое письмо и сложный объемный документ, яркую поздравительную открытку или рекламный блок.

По своим функциям Word вплотную приближается к издательским системам и программам верстки. Это значит, что в этом редакторе можно полностью подготовить к печати (или, как говорят специалисты, сверстать) журнал, газету или даже книгу, изготовить WWW-страницу Интернет.

Программы, создающие и обрабатывающие текстовую информацию, можно разделить на:

— редакторы общего назначения (например, Блокнот, Editor);

— редакторы исходных текстов программ (встроенные редакторы систем программирования, например, Turbo Pascal, QBasic),

— редакторы научных документов (например, ChiWriter, Tex);

— процессоры общего назначения (например, Microsoft Word); издательские системы (например, Adobe PageMaker, Corel Ventura) Что умеет Microsoft Word?

- Возможность создания нового документа с помощью специальных шаблонов (в частности, в Word включены шаблоны стандартных писем, поздравительных записок, отчетов, факсов и ряд других офисных доку- ментов).
- Возможность одновременного открытия и работы с большим количеством документов.
- Автоматическая проверка орфографии, грамматики и даже стили- стики при вводе документа.
- Автоматическая коррекция наиболее часто повторяющихся ошибок.
- Расширенные возможности форматирования документа. Допускает выравнивание документа по обоим краям, многоколоночную верстку.
- Использование стилей для быстрого форматирования документа.
- Возможность автоматизации ввода повторяющихся и стандартных элементов текста.
- Удобные механизмы работы с ссылками, сносками, колонтитула- ми.
- Включение в текст элементов, созданных в других программах
- Microsoft Office, — графических изображений, электронных таблиц и графи- ков, звуков, видеоизображений и так далее.
- Возможность подготовки простых электронных таблиц и гипер- текстовых документов Интернет.
- Возможность работы с математическими формулами.
- Возможность автоматического создания указателей и оглавления документа.
- Возможность отправки готового документа непосредственно из Microsoft Word на факс и по электронной почте (в обоих случаях необходимо, чтобы компьютер пользователя был оснащен модемом).
- Расширенные возможности индексации готового документа.
- Встроенный Мастер подсказок и объемная система помощи.

...И многое, многое другое....

Прежде чем пользоваться Word, необходимо запустить его. Для это- го в главном меню выбрать.

Все приложения, открыть Microsoft Office и выбрать Microsoft Word (см. рисунок 1).

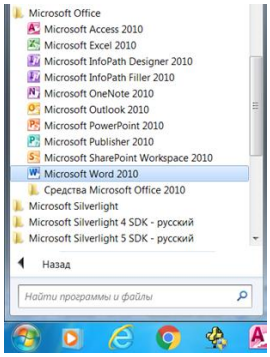


Рисунок 1 – Окно запуска Microsoft Word

К основным понятиям текстового процессора относятся:

Документ – совокупность объектов, разработанных для представления общей темы. Объектами могут быть фрагменты текста, рисунки, диаграммы, таблицы, видеоклипы и т.п.

Фрагмент – непрерывная часть текста.

Символ – минимальный элемент текста документа.

Слово — произвольная последовательность символов, ограниченная с обеих сторон служебными символами (пробелом, дефисом, точкой, запятой и т. п.).

Строка — произвольная последовательность символов между левой и правой границей абзаца.

Абзац – фрагмент текста, процесс ввода которого закончился нажатием клавиши <Enter>.

Страница — последовательность символов, ограниченная параметрами страницы документа.

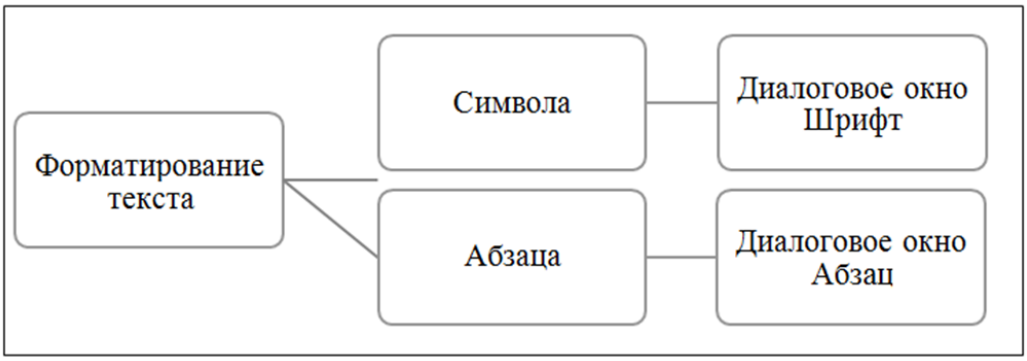


Рисунок 2 – Структура форматирования текста

Шрифт — набор символов, внешний вид которых единообразен.

Способ формирования шрифтов:

— Растровый — кодирование точек битовой матрицы. Растровые шрифты хранятся в файлах с расширением .fon.

— Векторный — расчет контуров символов по определенным формулам. Векторные шрифты хранятся в файлах с расширением .ttf.

Характеристики шрифтов:

— кегль — размер шрифта (1 пункт = 1/72 дюйма = 0,353 мм, 1 дюйм = 2,54 см);

— гарнитура — вид шрифта, одного рисунка (например, Times New Roman, Arial, Courier);

— начертание (обычный, полужирный, курсив, полужирный курсив);

— ширина (пропорциональный, моноширинный).

Стиль — определенное оформление, присущее абзацу текста. Основными элементами стиля являются:

— гарнитура, кегль и начертание шрифта;

— интерлиньяж (расстояние между строками);

— метод выравнивания (по левому или по правому краю, по центру, по ширине);

— красная строка, абзацный отступ слева и справа;

— интервал между абзацами.

Шаблон – заранее отформатированный текст; например, шаблон приказа.

Буфер обмена – часть оперативной памяти, используемой для временного хранения информации.

Этапы работы над текстовым документом:

1. Форматирование страницы

Форматирование страницы устанавливается с помощью диалогового окна Параметры страницы (см. Рисунок 3), вызываемого двойным щелч- ком по полю горизонтальной линейки.

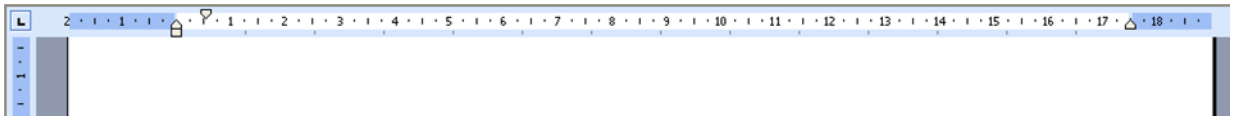


Рисунок 3 – Линейка

Горизонтальную линейку можно вывести на экран посредством ко- манды Вид → Линейка. Серая часть горизонтальной линейки отвечает за поля (левое и правое). С помощью линейки и обитающих на ней бегунков можно установить ширину текста и величину абзацного отступа. Нижние бегунки – правый и левый – отвечают за отступ основного текста от границ страницы (границы выделены серым цветом), а верхний бегунок – за абзацный отступ.

При форматировании страницы устанавливаются поля (см. Рисунок 4), ориентация страницы (книжная, альбомная), размер листа (для деловой документации в соответствии со стандартом СТО 005-2015, лист формата А4 с размерами 21×29,7 см), установлен раздел для колонтитулов (для верхнего 0, для нижнего под нумерацию страницы 1,25 см).

3. Общее форматирование для дальнейшего редактирования текста устанавливается:

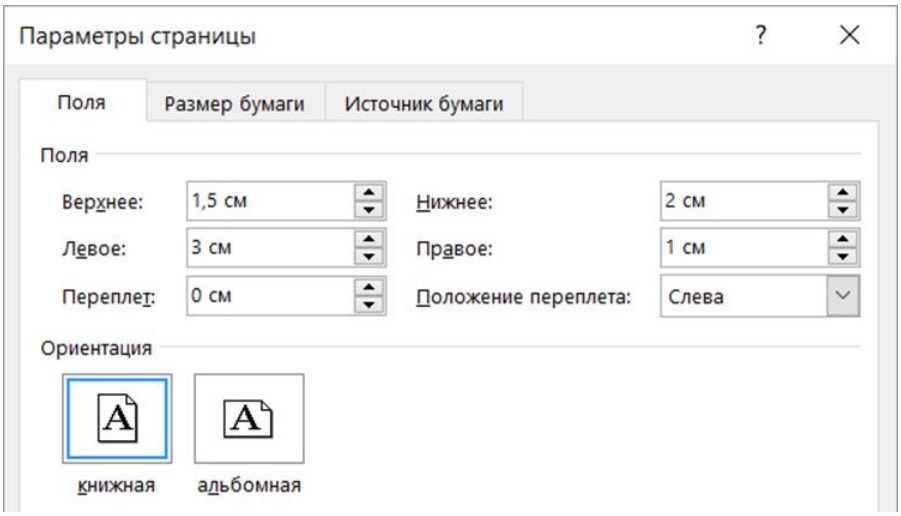


Рисунок 4 – Диалоговое окно Параметры страницы с установленными полями в соответствии со стандартом 005

Для форматирования символов: шрифт Times New Roman, размер – 14 пт, начертание – обычное, цвет – авто (черный).

Для форматирования абзацев: выравнивание – по ширине, отступ (слева, справа) – 0 см, первая строка (начало абзаца) – отступ 1,25 см, интервал (перед, после) – 0 см, межстрочный интервал – одинарный. Удобнее выставить все параметры сразу в диалоговом окне Абзац, вызываемый с помощью контекстного меню (правая кнопка мыши или групповой кноп- кой в соответствующем разделе на вкладке Главная) (см. Рисунок 5).

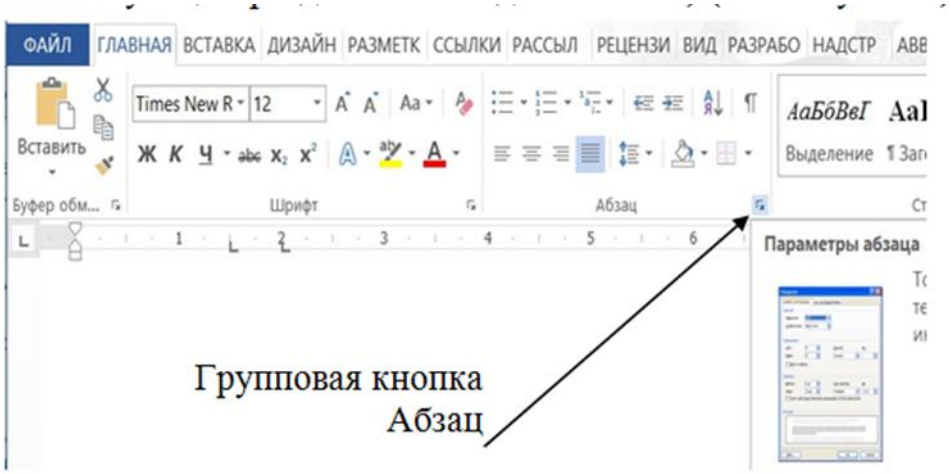


Рисунок 5 – Вызов диалогового окна Абзац с помощью групповой кнопки

1. Первичное сохранение документа через Файл - Сохранить как…
2. Редактирование текста

Редактирование текста включает ввод, перемещение, копирование и вставка фрагмента текста, проверка орфографии.

При использовании клавиатуры используется следующая комбинация:

<Ctrl>+<C> - копировать,

<Ctrl>+<X> - вырезать,

<Ctrl>+<V> - вставить.

Для проверки орфографии нужно подать команду Рецензирование Правописание.

При редактировании текста в первую очередь необходимо выделить фрагмент.

Таблица 1 - Способы выделения текста

Что выделить	Действие
Слово	Дважды щелкнуть на этом слове
Предложение	Удерживая клавишу Ctrl щелкнуть в любом месте предложения
Абзац	Трижды щелкнуть в любом месте абзаца или 2 раза по левому полю напротив абзаца
Строка	Поместить указатель на левое поле; когда он примет вид стрелки, указывающей вправо, выбрать строку и щелкнуть
Весь документ	Ctrl+A
Несколько строк	Переместить указатель к левому полю. Стрелкой, указывающей вправо, выбрать первую строку и щелкнуть. Не отпуская кнопку мыши, протащить ее рядом с выделяемыми строками

Несколько слов, строк или предложений	Поместить указатель в виде буквы 1 в начало первого слова, удерживая кнопку мыши, протащить ее до по-следнего нужного слова
Несколько слов, строк или предложений с помощью Shift-выделения	Щелкнуть на первом слове, переместить указатель кпоследнему слову (не удерживая кнопку), нажать Shiftи щелкнуть. Все, что находится между двумя щелчка- ми, будет выделено

Правила компьютерного набора текста:

— знаки препинания не отделяются пробелом от предшествующего текста (допускается отделять пробелом только вопросительный знак);

— знаки «процент», «градус», «минута», «секунда» от цифр не отделяются пробелом;

— цифры отделяются от № и § одним неразрывным пробелом;

— дефис не отделяется пробелом от предшествующего и последующего текста;

— тире (<Cr1>+минус на числовой клавиатуре) отделяется пробелом от предшествующего и последующего текста;

— слова, заключенные в кавычки или скобки, не должны отделяться от них пробелами;

— точки в конце заголовков не ставятся;

— после предлога, с которого начинается предложение, ставится неразрывный пробел;

— неразрывный пробел ставят при вводе дат (которые не принято располагать на двух строчках), фамилий с инициалами и т.п.;

— в конце общеупотребительных сокращений «кг», «т», «ц», «км» точка не ставится;

— сокращения типа «т.д.» и «с.г.» записываются без пробела;

— сокращения «и др.», «и т.д.», «и т.п.» могут применяться только в конце предложения, в середине предложения их нужно записывать полностью;

— там, где нужен пробел, но недопустим перенос на следующую строку, ставить неразрывный пробел;

— там, где нужен дефис и недопустим перенос на следующую строку, необходимо ставить неразрывный дефис;

— нажимать клавишу <Enter> только для начала нового абзаца.

3. Форматирование текста

4. Создание автосодержания

5. Сохранение изменений в документе под прежним именем командой Сохранить.

Практическая часть:

Задание №1: Отредактировать и отформатировать текст.

Работа начинается с форматирования страниц

1. Установите параметры страницы в соответствии с описанием на этапе 1 и рисунке 3 в последовательности работы над документом.

2. Выполнить второй и третий этап в последовательности работы над документом, сохранив под именем Word_Фамилия.doc в папку, указанную преподавателем.

3. Отредактировать текст письма, приведенный на Рисунок 1. Редактирование текста заключается во вводе текста и проверке орфограмм. При осуществлении ввода текста сразу применяются первичные элементы форматирования, установленные на втором этапе работы над документом.

4. Отформатировать, введенный текст:

Первых четыре строки выровнять по центру, предварительно для первой строки выставить «НЕТ»;

Замечание: при выравнивании по центру красная строка не устанавливается.

Выполнить шрифтом Verdana (этот шрифт и Arial часто используются для заголовков) с 15 пт размером;

Замечание: при установке размера символов необходимо воспользоваться полем ввода.

МАГАЗИН «Книги»
пр. Жемчужниковых, 12
664009 г. Иркутск
(8-3952) 54-23-09
9 августа 2013 г.
Ирина Драгоманова
Референт-переводчик
НИИ «Энергоатом»
Цветочный пр. 2-212
664004, г. Иркутск
Уважаемая Ирина Сергеевна!

Мы рады сообщить Вам, что по итогам опроса «Лучшая книга по геологии», проведенного нашим магазином, лучшей была признана книга Н.В. Короновского «Общая геология». Все читатели, которые подали свои голоса за эту книгу, награждаются ценными призами от нашего магазина. Вы можете зайти к нам в удобное для Вас время и получить свой приз.

Кроме того, предъявив это письмо, вы можете приобрести в нашем магазине любое количество изданий с 40% скидкой. Среди изданий, на которые мы особенно хотели бы обратить Ваше внимание, назовем Р.З. Сафиева. Химия нефти и газа. Нефтяные дисперсные системы: состав и свойства (часть 1). Учебное пособие. М.: РГУ нефти и газа им. И.М. Губкина. 2004, а также Геоинформатика. Толковый словарь основных терминов. Надеемся, что Вам понравятся эти книги.

В этом месяце мы проводим выставку-продажу книг по искусству «Красота и мир». Журнал «Искусство» посвятил этому событию такие строки:

В эпоху технологического и информационного бума особенно приятно видеть неослабевающий интерес мыслящей части общества к вечным культурным ценностям. Мы считаем, что настоящая выставка поможет заинтересовать и тех людей, которые до этого не уделяли особого внимания вопросам изящных искусств.

Искренне Ваш,
Кирилл Быстров
Директор магазина «Книги»

Рисунок 1– Текст документа на этапе редактирования

с. Название **Магазин «Книги»** записать с разрядкой в 4 пт (Групповой кнопкой Шрифт - вкладка - Дополнительно – Интервал- Разряженный);

Первая строка **«Магазин «книги»** сделать начертание полужирным.

2. Установите интервал перед датой 26пт, после 8пт, выполнив: вклад ка *Главная - групповая кнопка Абзац*.

Замените дату «9 августа 2013 г.» на сегодняшнюю, выполнив: *Вставка - Дата и время* (см. Рисунок 2).

Замечание: Для замены активизируйте режим *Отобразить все знаки* кнопкой ¶, расположенной на вкладке *Главная* в разделе *Абзац*, и выделите дату без знака конца абзаца ¶

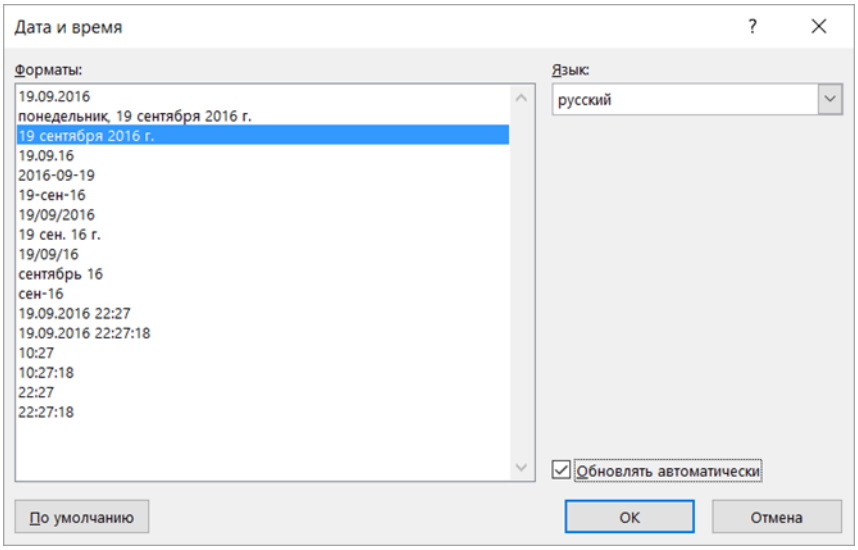


Рисунок 7 - Диалоговое окно Дата и время

1.Выберите пункт Русский (Россия) в раскрывающемся списке Язык (справа).

2. На экране появится окно диалога Дата и время. В нем показаны доступные форматы для даты и времени. Обратите внимание, что при этом вставляются текущие даты и время. Выберите формат, показанный в третьей строке сверху. Щелкните на кнопке ОК (Если понадобиться удалить дату, выделите ее с помощью мыши и нажмите Delete).

3. Для того, чтобы при каждом обращении к документу, дата обновлялась на текущую, необходимо установить опцию Обновлять автоматически.

4. Увеличить абзацный отступ (первая строка) до 2 см от даты до обращения «Уважаемая Ирина Сергеевна!».

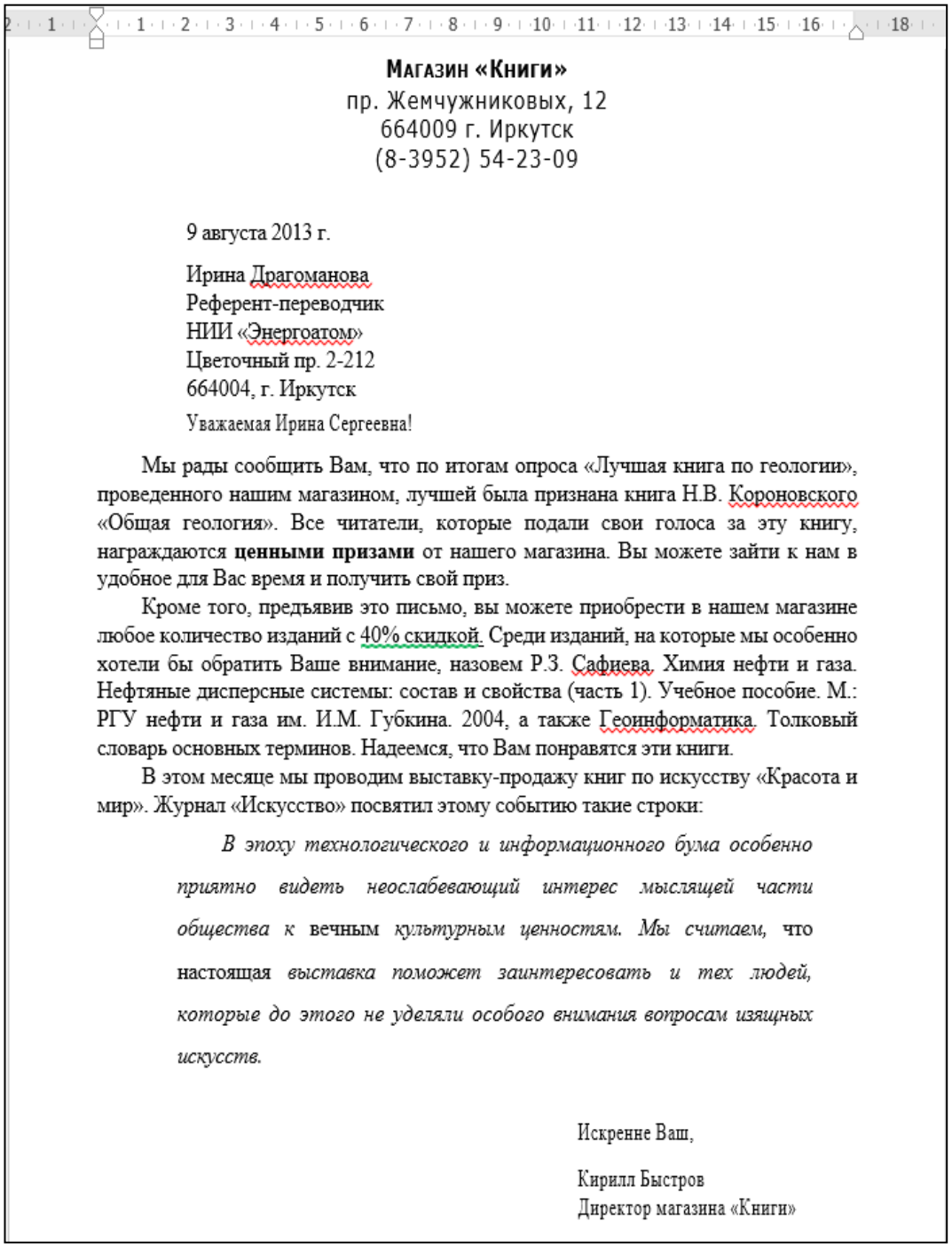
5. В обращении «Уважаемая Ирина Сергеевна!» установить интервал до и после обращения по 10 пт (самостоятельно), у самого обращения выставить Масштаб букв 70%.

6. Для цитаты установить курсив, межстрочный интервал равный 1,5 строки, отступ слева 2 см, справа 1 см.

7. В подписи установить отступ слева, как в образце, сделать масштаб букв 80% и для фразы «Искренне Ваш» установить интервал перед и после по 20 пт.

8. Выполнить индивидуальное форматирование в соответствии с образцом, представленном на рисунке 3

9. Сохранить изменения в файле.



Магазин «Книги»
пр. Жемчужниковых, 12
664009 г. Иркутск
(8-3952) 54-23-09

9 августа 2013 г.

Ирина Драгоманова
Референт-переводчик
НИИ «Энергоатом»
Цветочный пр. 2-212
664004, г. Иркутск

Уважаемая Ирина Сергеевна!

Мы рады сообщить Вам, что по итогам опроса «Лучшая книга по геологии», проведенного нашим магазином, лучшей была признана книга Н.В. Короновского «Общая геология». Все читатели, которые подали свои голоса за эту книгу, награждаются **ценными призами** от нашего магазина. Вы можете зайти к нам в удобное для Вас время и получить свой приз.

Кроме того, предъявив это письмо, вы можете приобрести в нашем магазине любое количество изданий с 40% скидкой. Среди изданий, на которые мы особенно хотели бы обратить Ваше внимание, назовем Р.З. Сафиева. Химия нефти и газа. Нефтяные дисперсные системы: состав и свойства (часть 1). Учебное пособие. М.: РГУ нефти и газа им. И.М. Губкина. 2004, а также Геоинформатика. Толковый словарь основных терминов. Надеемся, что Вам понравятся эти книги.

В этом месяце мы проводим выставку-продажу книг по искусству «Красота и мир». Журнал «Искусство» посвятил этому событию такие строки:

В эпоху технологического и информационного бума особенно приятно видеть неослабевающий интерес мыслящей части общества к вечным *культурным ценностям. Мы считаем,* что настоящая *выставка поможет заинтересовать и тех людей, которые до этого не уделяли особого внимания вопросам изящных искусств.*

Искренне Ваш,

Кирилл Быстров
Директор магазина «Книги»

Рисунок 3 – Отформатированный текст письма

Маркированные и нумерованные списки

Для увеличения большей наглядности текстового документа необходимо его структурировать. Одним из таких возможностей является создание **Списков.**

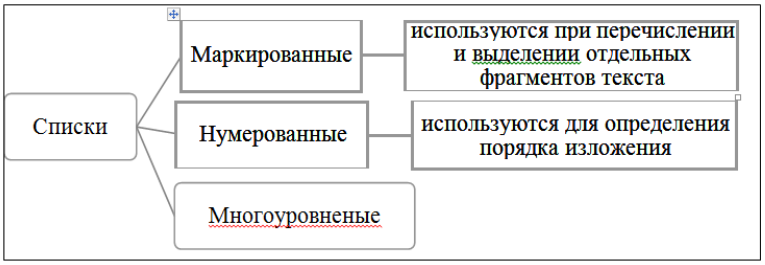


Рисунок 9 – Виды списко

Маркированные и нумерованные списки облегчают восприятие документа, отмечая ключевые моменты повествования или позволяя создавать нумерованные планы.

Маркеры и нумерацию можно добавлять в любое место документа, не прибегая к специальному форматированию. Если же необходимо отформатировать целый список элементов, то лучше воспользоваться функцией автоматического создания списков Word.

Задание №2: Создание маркированного списка.

Для создания маркированного списка (см. рисунок 4) можно использовать следующий способ: выбрать на вкладке: Главная → Маркеры → Выбрать маркер из списка или Определить новый маркер…

Рисунок 4 – Кнопки вызова списков на вкладке Главная.

В открывшемся диалоговом окне нажать на кнопку Символ… и выбрать нужный знак (разнообразие символов в шрифтах Webdings, Wingdings, Symbol), после этого в этом же диалоговом окне настраивается Шрифт маркера (см. рисунок 5)

Для создания маркированного списка выполните следующую последовательность действий:

1. Перейдите на новую страницу, нажав комбинацию клавиш Ctrl+Enter, введите номер задания и подзаголовок Списки, на следующей строке Маркированный список
2. Введите приведенный ниже текст:

К нефтегазовому оборудованию относятся:
Буровые установки;
Агрегаты для цементации и ГРП;
Оборудование для буровых установок;
Дуплексные насосы;
Триплексные насосы;
Буровые мотонасосные агрегаты;
Установка приготовления и очистки бурового раствора;
Трубопроводы, фасонные детали, арматура и др. оборудование;
Противовыбросовое оборудование;
Буровой инструмент.

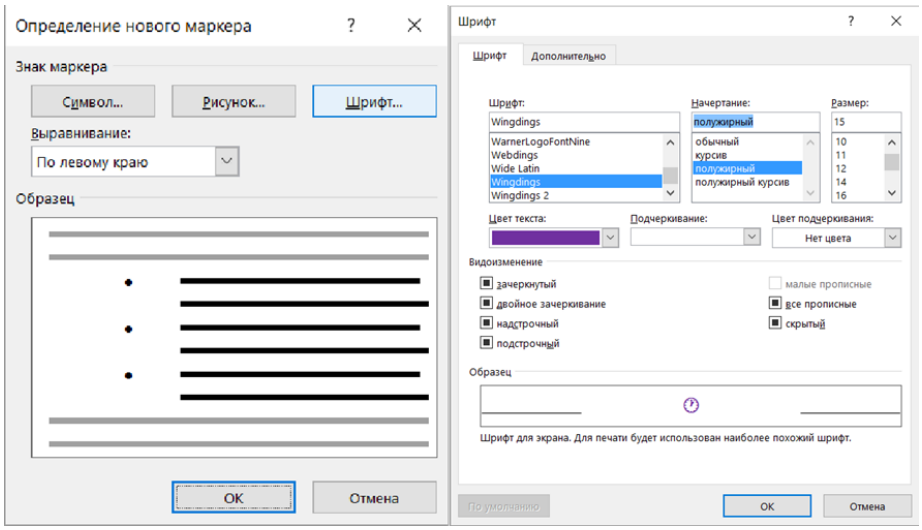
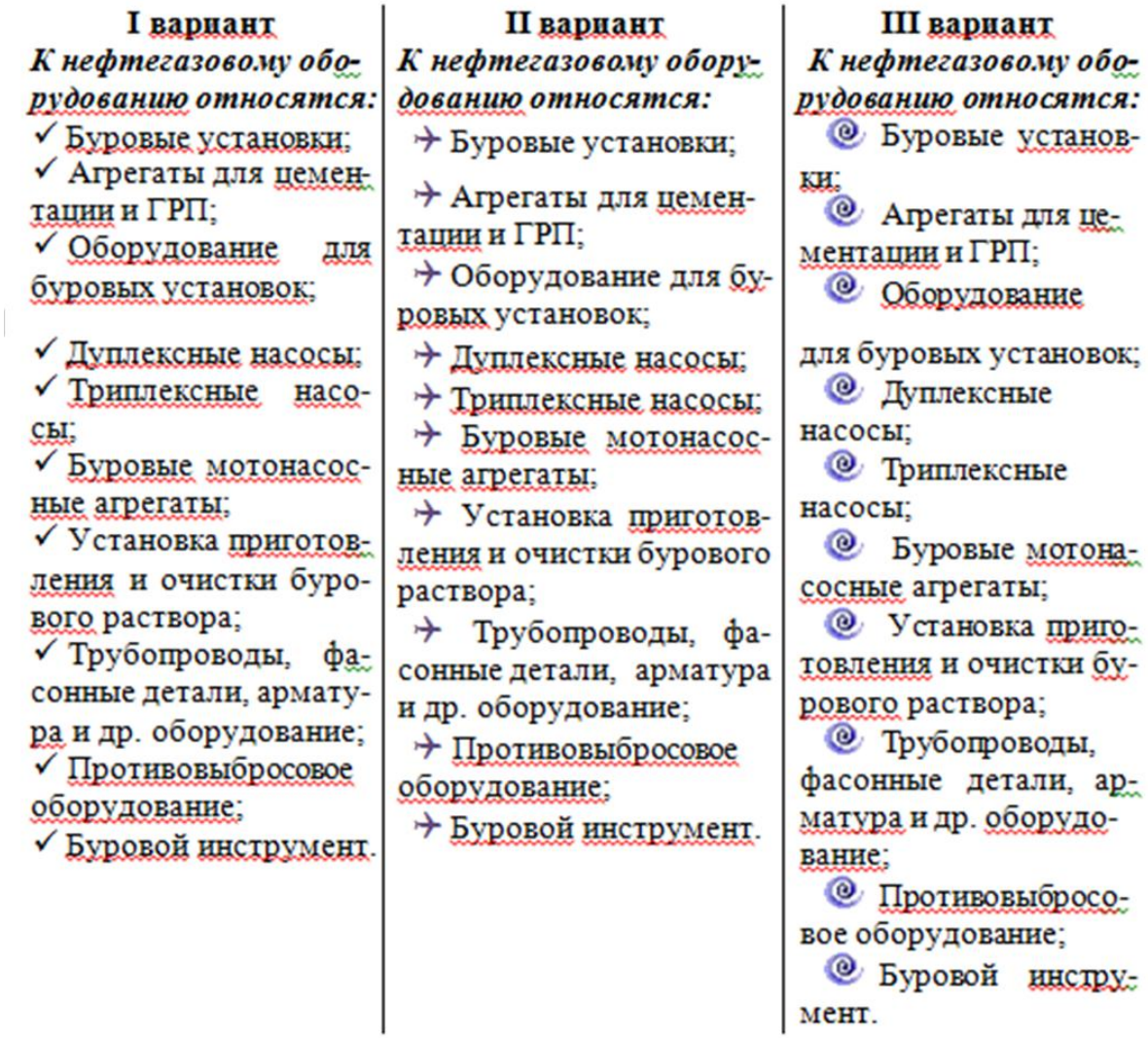


Рисунок 5 – Диалоговое окно форматирования символа

3. Скопируйте введенный текст и вставьте ниже восемь раз, в итоге должно быть девять копий;
4. Выделите первый фрагмент текста (без заголовка) и установите маркированный список, представленный ниже в первом варианте, на своё усмотрение воспользуйтесь кнопкой Шрифт… в диалоговом окне *Определение нового маркера* для установки размера, начертания, цвета символа;
5. Самостоятельно создайте список второго и третьего варианта, представленного ниже: для третьего варианта необходимо воспользоваться кнопкой Рисунок... и выбрать Локон, первоначально убрав настройки для второго варианта.



6. Для четвертого фрагмента придумать свой маркированный список с форматированием символа (самостоятельно).

Задание №3: Создание нумерованного списка

Параметры нумерованных списков аналогичны параметрам маркированных, но вместо маркеров в них используются номера.

Как и для маркированных списков, для создания нумерованных необходимо: выбрать на вкладке Главная → Нумерация → Выбрать номер из Библиотеки нумерации или Определить новый формат номера…, где выбирается нумерация, ставится Формат номера (символы которые повторяются (скобки, точки и т.п.), устанавливается выравнивание, после этого в этом же диалоговом окне настраивается Шрифт номера (см. рисунок 6).

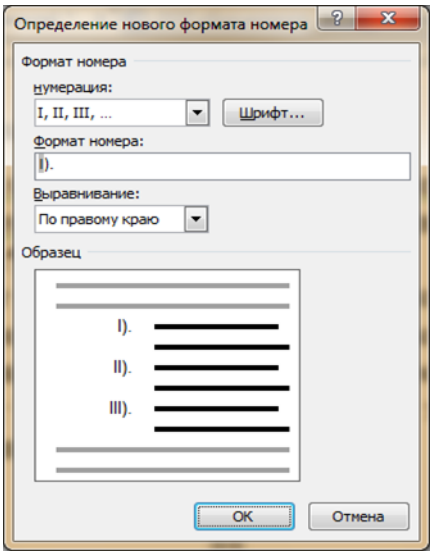


Рисунок 6 – Окно определения нового формата номер

Для создания нумерованного списка выполните следующую последовательность действий:

1. Установите курсор перед заголовком пятого фрагмента и перейдите на новую строку, введите название Нумерованный список;
2. Выделите пятый фрагмент текста (без заголовка) и установите нумерованный список, представленный ниже в четвертом варианте;
3. Самостоятельно создайте список пятого и шестого варианта, представленного ниже.

IV вариант	**V вариант**	**VI вариант**
К нефтегазовому оборудованию относятся:	***К нефтегазовому оборудованию относятся:***	***К нефтегазовому оборудованию относятся:***
i. Буровые установки; ii. Агрегаты для цементации и ГРП; iii. Оборудование для буровых установок; iv. Дуплексные насосы; v. Триплексные насосы; vi. Буровые мотонасосные агрегаты; vii. Установка приготовления и очистки бурового раствора; viii. Трубопроводы, фасонные детали, арматура и др. оборудование; ix. Противовыбросовое оборудование; x. Буровой инструмент.	1-й) Буровые установки; 2-й) Агрегаты для цементации и ГРП; 3-й) Оборудование для буровых установок; 4-й) Дуплексные насосы; 5-й) Триплексные насосы; 6-й) Буровые мотонасосные агрегаты; 7-й) Установка приготовления и очистки бурового раствора; 8-й) Трубопроводы, фасонные детали, арматура и др. оборудование; 9-й) Противовыбросовое оборудование; 10-й) Буровой инструмент.	!SAS! Буровые установки; !SBS! Агрегаты для цементации и ГРП; !SBS! Оборудование для буровых установок; !SГS! Дуплексные насосы; !SДS! Триплексные насосы; !SES! Буровые мотонасосные агрегаты; !SЖS! Установка приготовления и очистки бурового раствора; !SЗS! Трубопроводы, фасонные детали, арматура и др. оборудование; !SИS! Противовыбросовое оборудование; !SКS! Буровой инструмент.

4. Для восьмого фрагмента придумать свой нумерованный список с форматированием номера (самостоятельно).

Задание №4: Создание списка в документе

I. Предположим, что в целях повышения информативности документа было решено добавить названия некоторых доступных изданий в виде списка.

1. Скопируйте текст **Письмо о выставке**.

Уважаемые коллеги!

Приглашаем вас посетить нашу ежегодную выставку, которая состоится [дата] в [место/адрес]. Это уникальная возможность познакомиться с современными достижениями в области дизайна, искусства и культуры. В рамках мероприятия у вас будет возможность ознакомиться с работами лучших художников и дизайнеров, а также принять участие в специальных мастер-классах и презентациях.

Стоимость входных билетов — всего [цена] рублей. В честь открытия мы предлагаем скидку 40% на все билеты для первых 50 посетителей. Кроме того, при покупке билета онлайн вы получите дополнительный подарок.

Не упустите шанс получить новые впечатления и выгодные предложения! Для приобретения билетов и получения дополнительной информации обращайтесь по телефону [номер] или на наш сайт [сайт].

Ждём вас на выставке!

С уважением,

[Ваше имя]

[Ваша должность]

[Контактная информация]

2. Установите курсор перед «Уважаемые коллеги!» - выровняйте текс по центру
3. Выделите весь текст - Поставьте абзацный отступ 1,05
4. Щелкните после слов 50 посетителей . Нажмите клавишу Enter.
5. Выделите слова **-скидку 40% на все билеты для первых 50 посетителей, получите дополнительный подарок.**
6. Удалите конец предложения (до слов «Для приобретения» включительно).
7. Установите курсор в конец абзаца (после слова подарок). Нажмите Enter.
8. Введите свои данные после слова «С уважением», выровняйте текс по правому краю
9. Введите следующий текст представленный на картинке:

Яновская Т.Б., Порохова Л.Н. Обратные задачи геофизики.

Нажмите клавишу Enter.

Баландин Р.К. Энциклопедия драгоценных камней и минералов.

Нажмите клавишу Enter.

Светлана Гураль. Самые известные драгоценные камни. Путеводитель-определитель.

Нажмите клавишу Enter дважды.

После того как вы дважды нажмете клавишу Enter, маркированный список закончится. Теперь измените отступ этого списка, используя на линейке Отступ первой строки или используя Групповую кнопку Абзац.

II. Создайте список с указанием алгоритма проезда до выставочного павильона.

1. Щелкните в конце последнего абзаца (цитаты).
2. Нажмите клавишу Enter.
3. Нажмите клавиши Ctrl+Shift+N, чтобы восстановить форматирование абзаца по умолчанию.
4. Введите следующий текст:

Чтобы добраться до нашего магазина «Книги», следуйте по маршруту, предложенному ниже.

5. Щелкните в конце нового абзаца и нажмите клавишу Enter.
6. Введите следующий текст (без номеров).

От ст. метро «Академическая» по проспекту Науки до улицы Северной. Нажмите клавишу Enter.

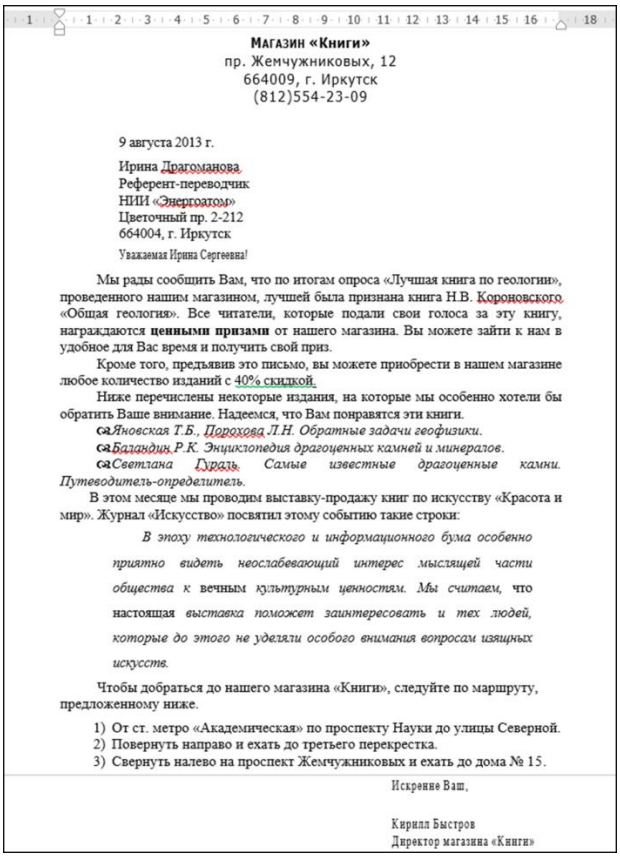
Повернуть направо и ехать до третьего перекрестка. Нажмите клавишу Enter.

Свернуть налево на проспект Жемчужниковых и ехать до дома №

Нажмите клавишу Enter.

7. Выделите три пункта списка и щелкните на кнопке Нумерация на вкладке Главная.
8. Измените параметры форматирования списка, выделив три пункта нумерованного списка.
10. Выберите команду Нумерация Определить новый формат номера.
11. В открывшемся окне диалога выберите вариант нумерации со скобками после чисел.

III. Теперь текст письма должен выглядеть так, как представлен на рисунке 7.



МАГАЗИН «КНИГИ»
пр. Жемчужниковых, 12
664009, г. Иркутск
(812)554-23-09

9 августа 2013 г.

Ирина Драгоманова
Референт-переводчик
НИИ «Энергоатом»
Цветочный пр. 2-212
664004, г. Иркутск

Уважаемая Ирина Сергеевна!

Мы рады сообщить Вам, что по итогам опроса «Лучшая книга по геологии», проведенного нашим магазином, лучшей была признана книга Н.В. Короновского «Общая геология». Все читатели, которые подали свои голоса за эту книгу, награждаются **ценными призами** от нашего магазина. Вы можете зайти к нам в удобное для Вас время и получить свой приз.

Кроме того, предъявив это письмо, вы можете приобрести в нашем магазине любое количество изданий с 40% скидкой.

Ниже перечислены некоторые издания, на которые мы особенно хотели бы обратить Ваше внимание. Надеемся, что Вам понравятся эти книги.

- *Яновская Т.Б., Дорохова Л.Н. Обратные задачи геофизики.*
- *Баландин Р.К. Энциклопедия драгоценных камней и минералов.*
- *Светлана Гураль. Самые известные драгоценные камни. Путеводитель-определитель.*

В этом месяце мы проводим выставку-продажу книг по искусству «Красота и мир». Журнал «Искусство» посвятил этому событию такие строки:

В эпоху технологического и информационного бума особенно приятно видеть неослабевающий интерес мыслящей части общества к вечным культурным ценностям. Мы считаем, что настоящая выставка поможет заинтересовать и тех людей, которые до этого не уделяли особого внимания вопросам изящных искусств.

Чтобы добраться до нашего магазина «Книги», следуйте по маршруту, предложенному ниже.

1) От ст. метро «Академическая» по проспекту Науки до улицы Северной.
2) Повернуть направо и ехать до третьего перекрестка.
3) Свернуть налево на проспект Жемчужниковых и ехать до дома № 15.

Искренне Ваш,

Кирилл Быстров
Директор магазина «Книги»

Рисунок 7 – Образец итогового документа

Задание №5: Создание колонтитулов

Колонтитулы — это надписи, помещаемые вверху и внизу каждой страницы документа. Они не относятся к основному тексту документа и обычно содержат информацию, которая либо помогает ориентироваться в многостраничных документах, либо служит для идентификации самого документа. В Word колонтитулами называются также сами области страницы, предназначенные для ввода такого текста.

В колонтитулы можно вставлять номера страниц, дату, время, имена файлов, фамилии, адреса, рисунки и многое другое.

Колонтитулы могут быть разными для разных фрагментов документа. Можно создать свои колонтитулы для четных и нечетных страниц, от- дельный колонтитул для титульного листа, а также разбить документ на разделы, для каждого из которых можно задать свои колонтитулы.

Добавьте в документ колонтитул, включающий номер страницы. Чаще всего колонтитулы используются именно для этого. Обратите внимание, что колонтитулы отображаются на экране только в режимах Раз- метка страницы и Предварительный просмотр.

1. Щелкните перед словами в этом месяце.
2. Нажмите клавиши Ctrl+Enter, чтобы вставить разрыв страницы.
3. Выберите команду Вставка→ Номер страницы Внизу страницы Простой номер 2. Вставьте в нижний колонтитул номер страницы.
4. Отформатируйте номер страницы шрифт Times New Roman, 12 пт, начертание обычное, выравнивание по центру без красной строки.

Номер страницы вставляется в колонтитул как поле, то есть в виде специального кода, который позволяет номерам автоматически обновляться при внесении изменений в документ.

Теперь на всех страницах документа будут располагаться номера страниц.

5. Номер страницы не ставится на титульный лист, чтобы он был невиден, необходимо на вкладке Работа с колонтитулами установить опцию (флажок) Особый колонтитул для первой страницы.

6. Вставить другую информацию в верхний колонтитул (его выбрать по своему вкусу) на вкладке Вставка Верхний колонтитул.

8. В левой части колонтитула, введите ФИО.

9. Поместите указатель мыши или текстовый курсор с помощью клавиши Tab в правую часть колонтитула и введите Группу.

15. Щелкните на кнопке Закрыть на вкладке Работа с колонтитилами.

16. Сохранить изменения в документе.

Контрольные вопросы:

1. Дайте определение «Колонтитул» - это…
2. Какой стандартный шрифт используется в Wodre?
3. Какой размер шрифта применяем в таблицах?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 9

Тема: Многостраничные документы

Количество часов на выполнение: 2

Цель работы: научиться основным приемам подготовки структурированного документа Microsoft Word; создание оглавления, внутренних и внешних ссылок. Отработать навыки форматирования готового многостраничного документа.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Многостраничный документ, для удобства подготовки и восприятия, должен иметь определенную структуру. Обычно такой документ имеет оглавление и содержательную часть, как любая книга. Для подготовки такого документа способ ручного ввода и корректировка оглавления имеет очевидные недостатки:

– ручное отслеживание всех изменений/добавлений названий

– заголовков в тексте и в оглавлении, нумерации страниц в оглавлении, трудоемко и сопряжено с ошибками;

– крайне неудобна навигация по электронному документу.

Создание оглавления. Microsoft Word 2010 содержит инструмент Ссылки/Оглавление/Оглавление, с помощью которого в документ можно легко вставить оглавление. Но, поскольку оглавление представляет список заголовков содержательной части, то сами заголовки должны быть предварительно обработаны с помощью встроенных в Microsoft Word форматов стилей заголовков (или стилей уровней структуры).

В Microsoft Word встроено девять различных стилей заголовков: Заголовок1 … Заголовок9.

Создание внутренних и внешних ссылок. Перемещение по тексту документа может быть организовано с помощью внутренних ссылок и закладок. Создание внутренней ссылки состоит из двух шагов: 1)Вставить закладку, 2)Установить гиперссылку на закладку.

Рецензирование документа. Под рецензированием здесь понимается внесение исправлений и примечаний в документ различными пользователями. При этом все сделанные поправки визуально выделяются и фиксируются в документе. В Microsoft Word 2010 режим исправлений включается через команду Исправления на вкладке Рецензирование в группе Запись исправлений.

По кнопке Показать исправления можно отображать различные элементы исправлений, исправления отдельных рецензентов, настраивать параметры исправлений.

По кнопке Исходный документ: Показать исправления можно посмотреть исходный документ, исходный документ с исправлениями или измененный документ, измененный документ с исправлениями.

К выделенному фрагменту текста можно добавить примечание:

Рецензирование /Примечания/ Создать примечание.

Практическая часть

Задание 1. Создать документ с оглавлением.

Методические указания:

1. Наберите текст.

Обучающие и оценочные ресурсы

Дистанционные курсы

… текст раздела 1 …

Виртуальные школы

… текст раздела 2 …

Дистанционные олимпиады, фестивали

… текст раздела 3…

Консультационные и презентационные ресурсы

Виртуальные педсоветы

… текст раздела 4 …

Проблемные форумы

… текст раздела 5 …

Методические ресурсы Минобразования

… текст раздела 6 …

Ресурсы образовательных учреждений

… текст раздела 7 …

Образовательные порталы компаний

… текст раздела 8 …

Информационные ресурсы

Электронные библиотеки и учебники
… текст раздела 9 …
Электронные каталоги и словари
… текст раздела 10 …
Электронные музеи
… текст раздела 11 …

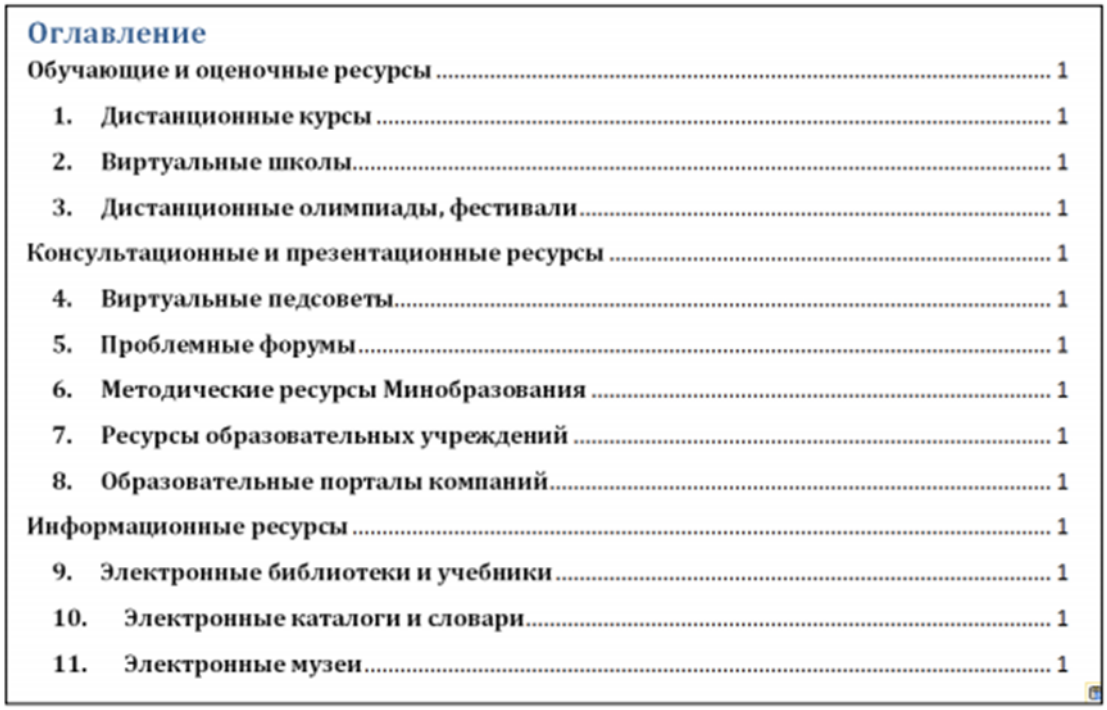
2. К заголовкам разделов (Обучающие …, Консультационные и…, Информационные ресурсы) примените стиль Заголовок1, а к заголовкам подразделов (Дистанционные курсы, Виртуальные школы) – стиль Заголовок 2 (Главная/Стили).

Действия по применению стилей заключаются в последовательном выделении заголовков текста, выборе и применении нужных стилей заголовков Заголовок 1, Заголовок 2. Используйте клавишу Ctrl для группового выделения объектов.

3. Для получения нумерованного списка к подразделам примените команду

4. Добавьте оглавление, предварительно поставив курсор в начало документа, выбрав пункт меню Ссылки /Оглавление/Оглавление.

В итоге будет создано оглавление (как в образце).



Примечание: если Вы добавляете в тексте новые заголовки (со стилями) или меняете старые, то следует выполнять обновление оглавления. Выделите оглавление, щелкните правой кнопкой мыши по нему и выберите пункт *Обновить поле*. Далее выберите *Обновить целиком*.

Есть простой способ копирования стилей, форматов для самых разных объектов *Microsoft Word* – кнопкой *Копировать формат*  : встаньте на отформатированный элемент и нажмите кнопку *Копировать формат*, а при установке курсора на другой элемент, к нему будет применен скопированный формат или стиль.

Навигация по документу с оглавлением реализуется с помощью гиперссылок, встроенных в оглавление. Удерживая клавишу *Ctrl*, Вы можете перейти на любой заголовок в тексте.

5. Используя оглавление, перейдите на подраздел **10. Электронные каталоги и словари.**

6. Измените различные атрибуты *Оглавления* – шрифт (Arial), нумерацию строк и пр.

Задание 2. Сформировать внутренние и внешние ссылки.

Методические указания:

1. Вставьте закладку:

- выбрать мышью место закладки, например, «…текст раздела 11…»;
- добавить имя этой закладки текст11 (*Вставка/Ссылки/ Закладка*).

Примечание: имя закладки должно состоять из одного слова, *наличие пробелов не допустимо.*

2. Установите гиперссылку на закладку:

- выделить текст или выбрать курсором место, откуда следует выполнять переход на закладку (например, «…**текст раздела 3…**»), и выполните команду *Вставка/Ссылки/Гиперссылка,* или в контекстном меню выбрать пункт *Гиперссылка;*
- в открывшемся окне выбрать опцию *Связать с местом в документе,* далее нужную закладку **текст 11** и нажать *Ок*. Тут же в поле *Текст* можно ввести текст всплывающей подсказки.

3. Вставить внешнюю гиперссылку, т.е. ссылку на Интернет страницу или файл.

- в разделе 6 введите текст **«ссылка на портал Минобразования РФ»** и выделите его.
- далее, щелкните правой кнопкой мыши по выделенному тексту, выберите пункт *Гиперссылка.* В открывшемся окне выберите опцию *Связь с файлом, веб-страницей*, в поле адрес введите www.mon.gov.ru и нажмите *Ок.* Переход по гиперссылке выполняется удерживанием клавиши *Ctrl.*

Задание 3. Выполнить рецензирование документа.

Методические указания:

1. Включите режим исправлений: *Рецензирование/Запись исправлений/Исправления*.

2. Название подраздела **7. Ресурсы образовательных учреждений**

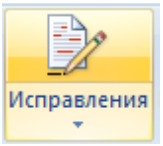
измените **на Ресурсы ВУЗов**.

3. К названию подраздела **Дистанционные олимпиады, фестивали** добавьте **и конференции**.

4. Создайте примечание к разделу **Обучающие и оценочные ресурсы**. В примечание введите «ФИО рецензента и № группы».

5. Посмотрите, как будет выглядеть документ после исправлений.

6. Посмотрите исходный документ, исходный документ с исправлениями или измененный документ, измененный документ с исправлениями: *Исходный документ/Показать исправления*.

7. Выключите режим исправлений, нажав на кнопку 

.

Задание 4. Выполнить форматирование готового документа.

Методические указания:

1. Вставьте титульный лист:

- выполните команду *Вставка/Страницы/Титульная страница* и выберите один из предложенных шаблонов;
- внесите свои данные.

2. Создайте разрывы страниц между разделами:

- встаньте перед названием первого раздела **Обучающие и оценочные ресурсы;**
- выполните команду *Вставка/Страницы/Разрыв страницы* или *Разметка страницы/Параметры страницы/Разрывы/Разрывы страниц/Страница*;
- аналогично создайте разрывы страниц для двух других разделов.

3. Добавьте ФИО, № группы и дату создания документа в верхний колонтитул:

- выполните команду *Вставка/Колонтитулы /Верхний колонтитул* и выберите из предложенного списка **Пустой (3 столбца);**
- в первый столбец наберите ФИО, во 2-ой столбец – № группы;
- для вставки даты в 3-ий столбец выполните команду *Работа с колонтитулами/Конструктор/Вставка/Дата и время,* выберите понравившийся формат;
- чтобы не было колонтитулов на титульном листе установите особый колонтитул для первой страницы: *Работа с колонтитулами/Параметры/Особый колонтитул для первой страницы* (установите флажок).

4. Вставьте номер страницы (*Вставка/Колонтитулы/Номер страницы/Внизу страницы*).

5. Добавьте новый раздел **Литература** со списком литературы.

- введите слово **Литература** и оформите стилем *Заголовок 1*;

- введите предложенный ниже список литературы и интернет-ресурсы:

1. Информатика. Базовый курс: учебник для вузов. 3-е изд. Стандарт третьего поколения. /Под ред. С.В. Симоновича. - СПб: Издательство «Питер», 2020.- 640 с. Ил.

2. Информатика. Компьютерный практикум: учебное пособие.-3-е изд., перераб. и доп./ под.ред. В.Т. Безручко - М.:ИД «Форум»: ИНФРА-М,2020.-368 с.: Ил.

Интернет-ресурсы:

1. http://wikipedia.org/wiki – свободная энциклопедия, которую может редактировать каждый.

2. http://www.yandex.ru – поисковая система.

6. Обновите оглавление.

Контрольные вопросы:

1. Как создать оглавление?
2. Для чего необходимо обновление оглавления?
3. Что такое внутренние и внешние ссылки?
4. Как отредактировать колонтитул?
5. Чем отличается постоянный колонтитул от переменного?
6. Как разбить документ на разделы?
7. Для чего необходимо рецензирование?
8. Как создать примечание в документе?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 10

Тема: Коллективная работа с документами

Количество часов на выполнение: 2.

Цель работы: уметь осуществлять редактирование документов нескольким пользователям и затем объединять сделанные исправления в исходном документе**.**

Оборудование: Персональный компьютер, MS Word

Методика выполнения задания

Теоретическая часть

Большое внимание следует уделить возможности коллективной обработки документов.

Текстовый редактор Word имеет средства, позволяющие редактировать документы нескольким пользователям и затем объединять сделанные исправления в исходном документе. К таким средствам относятся примечания, исправления и рассылка документов.

Примечания – это обозначенные инициалами и пронумерованные комментарии, которые записываются и отображаются в специальном окне примечаний и не затрагивают текст документа. Перед тем как вставить примечание, имеет смысл выделить фрагмент текста, который следует прокомментировать. В этом случае, при просмотре примечания текст, к которому оно относится, будет подсвечен.

Для вставки примечаний в текст документа нужно выполнить команды **Вставка/Примечание**; для просмотра примечаний **Вид/Примечания**.

Для отслеживания собственных исправлений, а также чтобы рецензенты могли вносить изменения в документ, текстовый редактор предлагает использовать маркеры исправлений. Пометка исправлений включается последовательностью команд **Сервис/Исправления/Выделить** исправления. Затем установить маркер в поле Записывать исправления. Маркеры исправлений помогают увидеть, какие изменения были внесены в документ по сравнению с его последней версией. Для изображения редактируемого текста используется специальный формат, например, подчеркивание. С помощью маркеров исправлений можно сохранить запись о каждом сделанном исправлении и в дальнейшем либо принять его, либо отказаться. Исправление помечается полным именем его автора, датой и временем создания.

Если во время редактирования документа маркер исправлений не был включен, то внесенные изменения можно найти, сравнив новую и старую версии документа. Два сравниваемых документа должны иметь различные имена файлов или различные местоположения. Для сравнения версий документа применяются команды **Сервис/Исправления/Сравнить версии**. На экране отображается отредактированный документ и вставленный, удаленный и исправленный текст с маркерами исправлений.

Для того чтобы разрешить пользователям комментировать текст без внесения изменений в документ, его нужно защитить с правом ввода примечаний. Возможность вносить изменения в документ позволит защита с правом ввода исправлений. Для максимальной эффективности защиты используется *пароль*.

Для установки защиты документа нужно выполнить команды **Сервис/Установить защиту** с указанием режима (запретить вставку примечаний, запись исправлений, ввод данных в поля форм). Защиту можно снять командами **Сервис/Снять защиту**, при необходимости задав пароль.

Если в редактировании документа принимают участие несколько пользователей, то имеется возможность объединить все исправления и примечания в исходный документ, для просмотра сразу всех изменений. В качестве исходного файла задается документ, в котором предполагается объединить исправления. Исправления, сделанные в документе, не могут быть объединены, если они не были помечены.

Чтобы слить исправления и примечания выполняются команды **Сервис/Исправления/Объединить исправления**.

Пользователь имеет возможность принять или отклонить сделанные исправления, выполнив команды **Сервис/Исправления/Принять/Отклонить исправления**.

При работе в текстовом редакторе с документами большого объема возникают определенные трудности, связанные с перемещением по документу, поиском информации, различной структурой разделов и др. По мере создания многостраничного документа следует определить его структуру - разбивку на главы, параграфы. Для общего представления о документе, быстрого перехода к нужному месту, организации и реорганизации заголовков и текста следует переключиться в режим просмотра структуры (**Вид/Структура документа**), где можно повысить или понизить уровень выделенных заголовков, поменять местами, изменить стили. Перечисленные действия выполняются с помощью панели инструментов, которая появляется при просмотре структуры.

Коллективная работа с документами строится на технологиях *groupware* и *workflow*[5].

Технологии groupware ориентированы на небольшие рабочие группы и характеризуются поддержкой выполнения одной коллективной задачи и отсутствием структуризации в организации работ. Поддержка ограничивается обеспечением коллективного доступа к информации с помощью различных *способов доступа*:

- сетевой доступ к файлам и базе данных;
- локальная и глобальная электронная почта (включая конференции и дискуссии);
- терминальный доступ, пересылка файлов и электронная доска объявлений;
- просмотр и интерпретация гипертекста (гипермедиа).

Технологии workflow служат для автоматизации документооборота в средних и крупных офисах и для них характерно:

- поддержка многопользовательской работы с несколькими задачами одновременно;
- четкая структуризация выполнения работ по ролям и документам с контролем исполнения.

Деловой процесс формализуется как совокупность состояний и переходов, необходимых для описания взаимодействия, как минимум двух субъектов (в частном случае сотрудников предприятия) для достижения выполнения заранее заданного условия. Частным случаем такого взаимодействия является простая пересылка документа из точки в точку.

Одной из реализаций технологии workflow является так называемая "система графов", где каждый шаг отражает движение задания, связанного с документом, или просто передвижения документа от одного субъекта к другому. При этом на человека, отвечающего за правильность функционирования схемы, ложится ответственность учета всевозможных непредвиденных (или отказных) ситуаций, которые могут возникнуть на пути движения документа.

Другая реализация основывается на понятии "цикл". В этом случае подразумевается, что наименьшим элементом в схеме взаимодействия является цикл, учитывающий взаимодействие между двумя произвольными субъектами. При этом система сама отслеживает правильность протекания процесса и в случае ошибки показывает место некорректности с указанием её причины, после чего прекращается воспроизведение нового процесса.

Установление правил взаимоотношений субъектов документооборота дополняется заданием безусловной и условной маршрутизация документов (по электронной почте) и времени обработки документа для контроля и учета исполнения.

Вывод информации

Вывод информации осуществляют путем печати документов, публикации их на Web- серверах, в общих почтовых папках и электронных досках объявлений или

рассылки по телекоммуникациям.

Обзор средств автоматизации учреждений

Информационно-программные средства автоматизации учреждений делятся на следующие категории:

- функциональные и интегрированные пакеты офисной автоматизации;
- системы для организации групповой работы;
- системы управления электронными документами;
- средства управления документооборотом.

Средства офисной автоматизации и коллективной работы в сети

Одним из наиболее распространённых средств офисной автоматизации является описанный выше интегрированный пакет Microsoft Office.

Для разработки и размещения прикладных программ, ориентированных на совместное использование информации, предназначена система *Lotus Notes,* представляющая собой платформу типа клиент-сервер. Система Lotus Notes позволяет пользователям получать, отслеживать, совместно использовать и создавать информацию, предназначенную для документов. Эта информация может поступать в различных форматах, таких как тексты, изображения, видео и звук, и от различных источников, таких как компьютерные прикладные системы, оперативные системы, сканеры или факс-аппараты. Пользователям система Lotus Notes обеспечивает доступ к сети через любой применяемый ими графический пользовательский интерфейс

Практическая часть:

Задание 1.

Указание. Работу по созданию текстового документа выполнять в команде.

1. Открыть текстовый редактор

Создать ⇒ Документ

2.Ознакомиться с интерфейсом и основными возможностями текстового редактора.

3. Задать поля страницы сверху, снизу, справа, слева по 2 см

меню Файл ⇒ Настройки страницы

4. Задать имя документу «Групповая работа»

меню Файл ⇒ Переименовать

5. Разрешить доступ к документу в качестве Редактора другим участником группы

Кнопка: Предоставить доступ ⇒ ввести электронные адреса участников команды⇒ выбрать уровень доступа Редактор ⇒ Открыть доступ ⇒ Готово

Либо

Предоставить ссылку участникам группы.

6. Записать всех участников группы

7. Создать таблицу, как показано ниже

База данных -	
Видеопамять –	
Графический редактор –	
Драйвер –	
Интерфейс –	
Компьютерный вирус –	
Оперативная память –	
Печатный документ –	

- Назвать таблицу «Понятия в информатике»

- Каждому понятию дать описание (каждый студент по 2)
- Добавить две строки и дать определения понятиям ***файловый сервер*** и ***шаблон.***

Другой способ для сетевого сбора информации от множества участников в таблицах Google – предоставление доступа к документу для заполнения формы. В этом случае ответы участников автоматически добавляются в электронную таблицу.

3. Оборудование: персональный компьютер с выходом в Интернет.

4. Порядок выполнения заданий.

1.Создайте текстовый документ на сервере docs.google.com на тему Планеты солнечной системы. Требования: шрифт – Nimes New Roman, 14пт, красная строка – 1,25, выравнивание – по ширине.

Каждая группа берет по одной теме: Меркурий, Венера, Земля, Марс, Группы астероидов, Церера, Юпитер, Сатурн, Уран, Нептун, Девятая планета. Каждую тему начинать с нового листа. Информацию брать с сайта https://ru.wikipedia.org. Наличие изображений обязательно.

Потренируйтесь в обсуждении размещённых документов и в добавлении комментариев внутри текста документов.

2. Создайте общую таблицу списка студентов вашей группы. Внесите в нее следующие данные: ФИО, возраст, рост, вес. Подготовьте диаграммы, наглядно демонстрирующие результаты деятельности. В сетевом офисе диаграммы и графики строятся так же, как и MS Office или Open Office.

3. Создайте для коллективного редактирования презентацию Планеты солнечной системы, используя информацию из текстового документа, созданного ранее.

Контрольные вопросы

1. Перечислите преимущества Интернет-офиса Google документ
2. В чем отличие прав пользователей Соавтор и Читатель при совместной работе в Интернет офисе
3. Перечислите виды документов с которыми Google Документы позволяет работать пользователям прямо в окне браузера

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 11

Тема: Обработка цифровых фотографий (кадрирование, исправление перспективы, коррекция уровней, коррекция цвета). Многослойные изображения.

Количество часов на выполнение: 4.

Цель работы: Изучение компьютерной графики и ее видов. Графический редактор Gimp.

Оборудование: Персональный компьютер,

Методика выполнения задания

Теоретическая часть

Виды компьютерной графики

Несмотря на то, что для работы с КГ существует множество классов программного обеспечения, выделяют следующие виды КГ:

растровую, векторную, фрактальную и трехмерную графику.

Они различаются принципами формирования изображения при отображении на экране монитора или при печати на бумаге.

Растровая графика применяется при разработке электронных и полиграфических изданий. Иллюстрации, выполненные средствами растровой графики, редко создают вручную с помощью компьютерных программ. Чаще для этой цели сканируют иллюстрации, подготовленные художником на бумаге, или фотографии. В последнее время для ввода растровых изображений в компьютер нашли широкое применение цифровые фото- и видеокамеры. Соответственно, большинство графических редакторов, предназначенных для работы с растровыми иллюстрациями, *ориентированы не столько на создание изображения, сколько на их обработку*. В Интернете, в основном, применяются растровые иллюстрации.

Программные средства для работы с *векторной графикой* наоборот предназначены, в первую очередь, *для создания иллюстраций и в меньшей степени для их обработки*. Такие средства широко используют в рекламных агентствах, дизайнерских бюро, редакциях и издательствах. Оформительские работы, основанные на применении шрифтов и простейших геометрических элементов, решаются средствами векторной графики намного проще. Существуют примеры высокохудожественных произведений, созданных средствами векторной графики, но они скорее исключение, чем правило, поскольку художественная подготовка иллюстраций средствами векторной графики чрезвычайно сложна.

Программные средства для работы с фрактальной графикой предназначены для автоматической генерации изображений путём математических расчетов.

Создание фрактальной художественной композиции состоит не в рисовании или оформлении, а в программировании. Фрактальную графику редко применяют для создания печатных или электронных документов, но ее часто используют в развлекательных программах.

GIMP — это мощный профессиональный графический редактор с массой вспомогательных программ. Само название «GIMP» является аббревиатурой GNU Manipulation Image Program и переводится на русский язык как «программа обработки изображений».

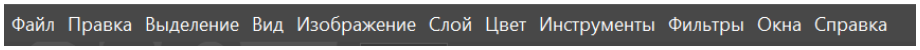
Основными задачами, почему стоит обратиться к программе GIMP, являются кадрирование и редактирование изображений. Посредством актуальных инструментов можно легко устранить недочеты заваленного горизонта, эффект красных глаз, неправильную экспозицию. Также редактор позволяет создавать из фотографий художественные композиции или дизайнерские коллажи различной сложности.

На главном окне программы находятся:

1. Строка меню
2. Панель Инструментов
3. Панель Цветов
4. Панель, Градиентов
5. Панель, Шаблонов и Кистей.

Главное меню

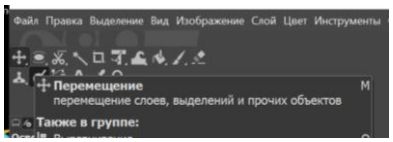
В самом верху окна располагается главное меню с такими вкладками: Файл, Правка, Выделение, Вид, Изображение, Слой, Цвет, Инструменты, Фильтры, Окна и Справка. В каждой из этих вкладок располагаются другие функции, которые можно использовать для работы.



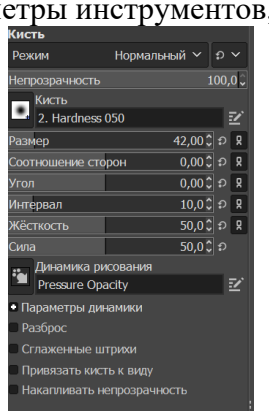
Панель инструментов

В левой стороне экрана, в его верхней части, располагается панель инструментов, в которой можно выбрать все необходимые для обработки графического файла инструменты: перемещение, поворот, вырезание, дублирование, карандаш, кисть, ластик, заливка, штамп, текст, а также многие другие.

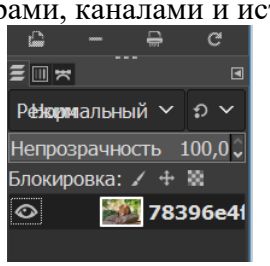
Если подвести курсор мышки к одной из кнопок на панели инструментов, появится всплывающая подсказка, рассказывающая о назначении инструмента.



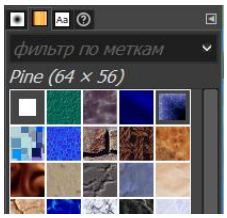
Двойной щелчок левой клавишей мыши по кнопке инструмента открывает окно Параметры инструментов, в котором можно задать параметры инструмента.



В правой части экрана вверху располагается панель управления слоями, контурами, каналами и историей ваших изменений.



можно выбирать различные кисти, градиенты, а также текстуры.



Окно изображения

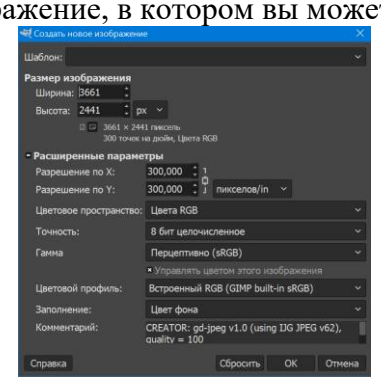
GIMP позволяет одновременно редактировать множество изображений. При этом каждое изображение открывается в отдельном рабочем окне, которое называется Окном изображения

Работа с файлами

Работа с файлами изображений осуществляется через пункт меню Файл. В редакторе GIMP есть два меню Файл: одно находится в панели инструментов, другое вы найдете в контекстном меню Окна. (вызывается левой кнопкой мыши по пустому полю).Эти меню несколько отличаются друг от друга.

Создание новых изображений

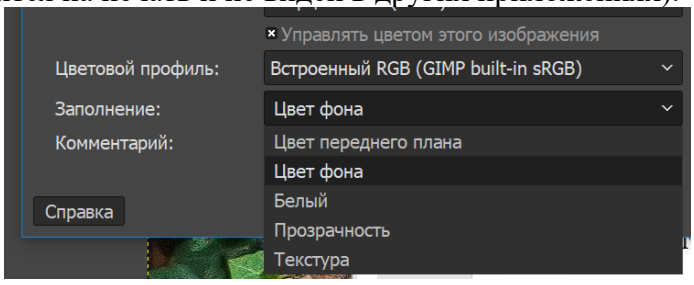
Чтобы начать редактировать изображение "с нуля", отдельно отредактировать часть изображения или сохранить часть изображения в новый файл, необходимо создать новое изображение, содержащее только фон. Его можно сравнить с чистым холстом. Для создания нового изображения выберите в меню Файл команду Новый, или нажмите комбинацию клавиш Ctrl+n. В результате появится окно Новое изображение, в котором вы можете задать размеры, тип и цвет фона изображения.



Используя **переключатель Заполнение**, задайте фон изображения:

- переднего плана - фон цвета, совпадающего с основным цветом;
- фон - фон цвета, совпадающего с фоновым цветом;
- белый - белый фон;
- прозрачное - прозрачный (бесцветный) фон.
- текстура

Прозрачный фон целиком состоит из прозрачных пикселов, для отображения которых редактор GIMP использует шаблон в виде шахматной доски (этот шаблон не выводится на печать и не виден в других приложениях).



Открытие изображений

Для того, чтобы открыть уже существующее (записанное в файл) изображение, выберите в меню Файл команду Открыть или нажмите комбинацию клавиш Ctrl+o. В результате появится окно Загрузка изображения, посредством которого можно осуществить выбор необходимого файла.

Сохранение изображений

Для того, чтобы сохранить изображение в файл, воспользуйтесь командой «Сохранить» меню «Файл» окна изображения. Для этого выберите в контекстном меню Файл команду «Сохранить», либо нажмите комбинацию клавиш Ctrl+s. Если изображение сохраняется впервые, появится окно «Сохранить изображение». В нем следует указать полный путь, имя и формат файла.

Сохранение изображений в формате JPEG

Если необходимо выбрать другое расширение, то воспользуйтесь – Файл/Экспортировать как…

Двойным щелчком по типу файла – выбираете нужный формат.

При сохранении изображения в формате JPEG на экране появится окно Сохранить как Jpeg, которое позволяет задать уровень качества, степень сглаживания, оптимизацию изображения и некоторые другие параметры.

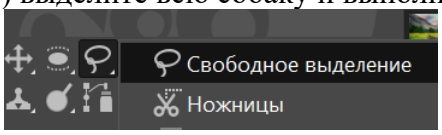
Сглаживание. При желании вы можете задать степень сглаживания, применяемого для корректировки резких границ, которые могут возникнуть в изображении в результате сохранения в формате JPEG.

Практическая часть:

Задание 1. Вырезание фрагмента фотографии и вставка его в другое фото

1. Откройте фото «пейзаж» и «хаски».

2. Выберите фото «хаски» и с помощью инструмента «Свободное выделение» (Лассо) выделите всю собаку и выполните обрезку.



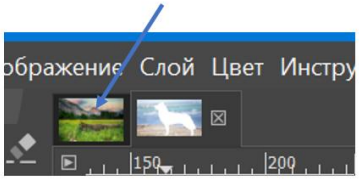
Для увеличения масштаба фотографии можно воспользоваться горячими клавишами Shift и +, для уменьшения Shift и –

Одновременное нажатие пробела и ЛКМ – можно двигать фото.

Выделяем полностью собаку, обводя весь ее контур. После того, как контур замкнули, вырезаем собаку Ctrl+X

Выбираем инструмент перемещения 

Далее выбираем наш фон с пейзажем, щелкнув по нему

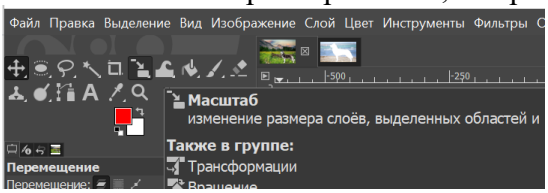


Создаем новый слой

Shift+Ctrl+N или в меню Слой/Создать новый слой

И вставляем собаку Ctrl+V

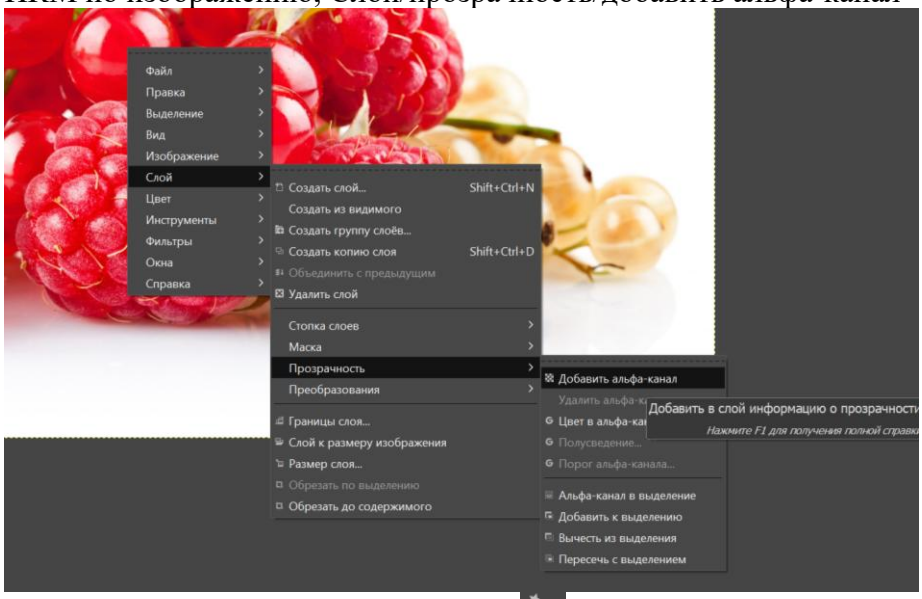
Можно поменять размер собаки, выбрав инструмент масштабирования



Задание 2. Вырезаем белый фон

Открыть файл «ягоды»

ПКМ по изображению, Слой/прозрачность/добавить альфа-канал



Инструментом «волшебная палочка»  выделяем белую область, нажимаем del

Берем ластик  и убираем лишние оставшиеся части

Создаем новый слой и заливаем его любым цветом при помощи инструмента «Заливка» 

Задание 3. Убираем ненужный объект с фотографии

Открыть файл «пляж»

Берем инструмент «Штамп» 

Зажимаем Ctrl и щелкаем по воде, как бы копируя текстуру

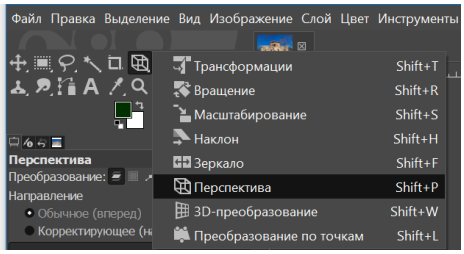
отпускаем Ctrl и начинаем стирать ненужный объект

Точно также удаляем лишнее на песке и небе.

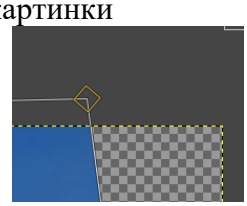
Если нужно подредактировать края, размыть, чтобы не было видно границ, можно воспользоваться инструментом «размазывание» 

Задание 4. Исправление перспективы

Откройте файл «пизанская башня». Выберите инструмент «перспектива» или нажмите Shift+P



Пробуя разные настройки этого инструмента, исправьте перспективу, потянув за края картинки



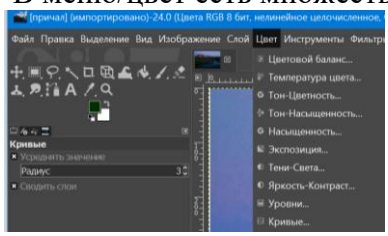
Выпрямите башню

Инструментом «кадрирование»  , отсеките ненужные части изображения

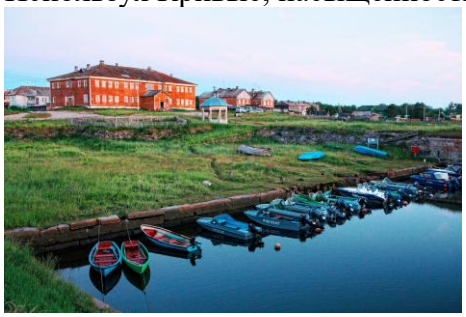
Задание 5. Коррекция цвета, тона, насыщенности

Откройте файл «причал», изображение очень темное

В меню/цвет есть множество настроек для работы с фотографией



Используя Кривые, насыщенность, цветовой баланс, исправьте фотографию



Задание 6. Многослойное изображение

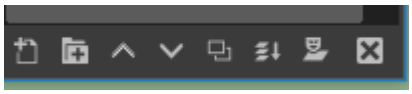
Понадобятся три файла: «закат», «люди» и «лайнер»

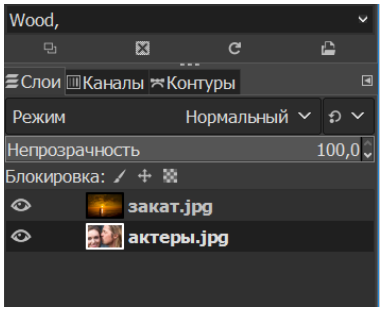
1. Открываем два файла «закат» и «люди» как слои. Меню/файл/открыть как слои. Зажав Ctrl выделяем обе фотографии и открываем.

Файлы должны открыться в одном документе, буду расположены один поверх другого.

2. Необходимо уменьшить в размере людей до слоя «закат», берем инструмент масштабирования или нажимаем Shift+S и уменьшаем.

Теперь необходимо переместить слой с парой вниз. Для этого наживаем кнопку «Опустить слой вниз».

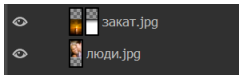


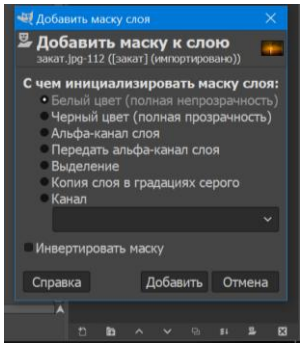


Если этот слой находится под слоем закат, то ничего тогда опускать не надо.

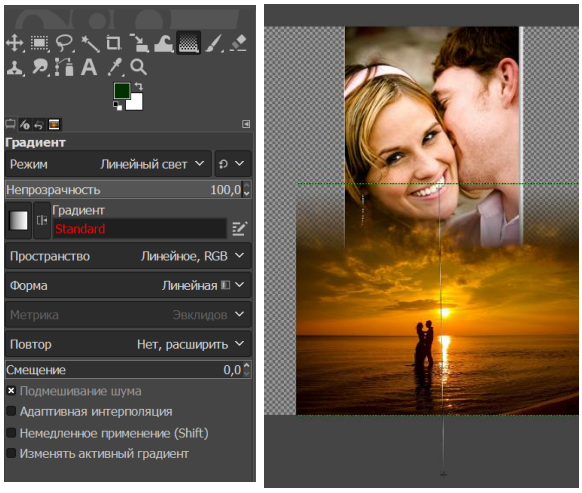
Слой «люди» находятся под слоем «закат». Меняем с помощью масштабирования размеры фото, С помощью инструмента «Перемещение» расположим слой с закатом, как на скриншоте

Переходим на слой с закатом и через контекстное меню слоя добавим маску белого цвета. 





Далее выбираем инструмент «Градиент» и проводим им по белой маске (тянем стрелочку по изображению сверху вниз)



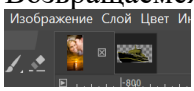
Кадрируем  , отсекаем все ненужное, чтобы ровное изображение было

3. Открываем файл «лайнер»,

Вырезаем белый фон, ПКМ по изображению, Слой/прозрачность/добавить альфа-канал

Инструментом «волшебная палочка»  выделяем белую область, нажимаем del

Возвращаемся к слою закат, щелкнув в меню по нему

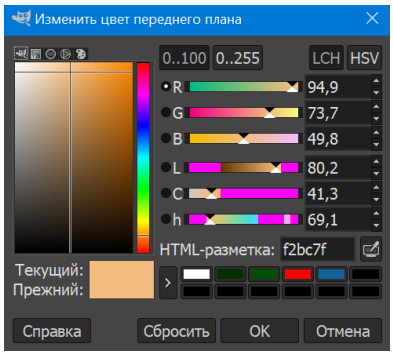


Создаем новый слой прозрачный (Ctrl+Shift+N) и вставляем корабль на этот слой, вырезав его или скопировав в том файле, где мы убирали белый фон. С помощью инструмента масштабирования (Shift+S) уменьшаем корабль, берем инструмент перетаскивания  и помещаем в угол

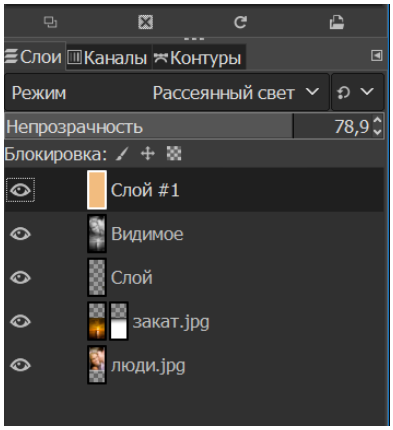
4. Создаем новый слой из видимого. Для этого выбираем верхний слой в стопке слоев, и вызвав контекстное меню ПКМ, выбираем пункт «Создать из видимого».

Обесцвечиваем данный слой через меню «Цвет/Обесцвечивание/ Обесцвечивание», режим «светимость», получится изображение в серых оттенках

Добавляем новый прозрачный слой (на самый верх) Ctrl+Shift+N и закрашиваем с помощью инструмента заливки  цветом — f2bc7f



Изменяем режим наложения текущего слоя на «Рассеянный свет» и с помощью ползунка регулируем непрозрачность до 70-80%.



Итог:

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.
4. Ответы на контрольные вопросы.
Практическая работа сохраняется в личной папке на сетевом диске.
Затем прикрепляется в Электронный курс https://el.istu.edu Информатика
Форма контроля: дифференцированная
Ссылки на источники:*[1]*

Практическая работа № 12

Тема: Векторная графика
Количество часов на выполнение: 2.
Цель работы: Ознакомиться с возможностями GIMP в работе с векторными объектами (через работу с путями). Научиться создавать и редактировать векторные элементы. Освоить основные инструменты и способы сохранения векторных данных в GIMP.
Оборудование: Персональный компьютер, GIMP.

Методика выполнения задания
Теоретическая часть

Теоретический материал
1. Что такое векторная графика?
Векторная графика — это способ представления изображений при помощи геометрических объектов: линий, кривых, многоугольников, окружностей и других фигур. Все элементы описываются математическими формулами, благодаря чему изображение сохраняет качество при изменении размера.
2. Отличия векторной графики от растровой:

Характеристика	**Векторная графика**	**Растровая графика**
Представление	Математические формулы, объекты	Пиксели (точки)
Масштабируемость	Без потери качества при увеличении	Потеря качества при увеличении
Размер файла	Меньше, зависит от сложности	Зависит от разрешения
Использование	Логотипы, схемы, чертежи	Фотографии, детали с множеством цветов

3. Основные форматы векторных изображений:
SVG (Scalable Vector Graphics)
AI (Adobe Illustrator)
EPS
WMF
4.Векторная графика в GIMP
GIMP — это в первую очередь растровый редактор, однако он обладает возможностями работы с путями, что позволяет создавать векторные элементы.
Пути в GIMP — это математические линии и формы, которые можно редактировать и масштабировать без потери качества.

5. Что такое путь?
Путь — это линия или кривая, созданная с помощью инструмента «Перо» (Paths Tool). Путями управляют узлами и кривыми, что дает возможность создавать точные геометрические фигуры.

6. Отличие путей от растровых объектов:

Пути — это векторные объекты, которые при необходимости можно залить цветом, заменить линию, изменить форму.

Пути позволяют сохранять точность при масштабировании.

7. Использование путей

Создание сложных форм

Обрезка изображений

Создание векторных элементов для дальнейших проработок

Задание:

Задание 1. Создание нового файла

Откройте GIMP.

Создайте новый файл (Файл → Новый).

Задайте размеры, например, 800x600 пикселей.

Задание 2. Создание и редактирование путей

Выберите инструмент «Путь» (Paths Tool) — иконка пера или через меню Инструменты → Пути.

Нарисуйте простую фигуру: к примеру, треугольник, кучу линий или овальную фигуру.

Щелкайте мышью, чтобы создавать узлы.

Чтобы сделать кривую, щелкните и перетащите узлы или кривые «ручки».

После создания пути его можно редактировать — смещать узлы, менять формы кривых, добавлять новые узлы.

Задание 3. Работа с путями

Для заливки созданного пути сделайте правый клик по пути в панели «Пути» и выберите «Залить путь».

Можно также выбрать «Обвести путь» — задать толщину и цвет линии.

Для редактирования формы перемещайте узлы, используйте инструмент «Касательная» («Move Point»).

Задание 4. Использование путей для маски и обрезки

Создайте слой с изображением или фоном.

Щелкните правой кнопкой по пути и выберите «Создать маску», чтобы вырезать или выделить часть изображения.

Можно сохранить полученный результат.

Задание 5. Сохранение и экспорт

Сохраните файл (.XCF) — исходный проект.

Для распространения используйте экспорт в форматы PNG или JPEG (Файл → Экспортировать As).

Задание 6

Создайте векторную фигуру (например, логотип или сложную геометрическую фигуру), используя инструмент «Путь», редактируйте узлы и кривые, создайте заливку или обводку, а затем экспортируйте результат.

Контрольные вопросы:

1. Чем отличаются пути в GIMP от объектов в полноценных векторных редакторах?
2. Какие инструменты позволяют редактировать узлы и кривые?
3. Почему важно сохранять работу в формате XCF?
4. Какие преимущества в использовании путей при создании сложных элементов?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 13

Тема: Интерактивные мультимедийные объекты на слайде

Количество часов на выполнение: 4

Цель работы: Ознакомиться с возможностями добавления интерактивных и мультимедийных элементов в презентацию. Научиться вставлять и настраивать мультимедийные объекты: звук, видео, гиперссылки, кнопки и анимацию. Создать интерактивный слайд с элементами управления.

Оборудование: Персональный компьютер, Microsoft PowerPoint

Методика выполнения задания
Теоретическая часть

1. Что такое мультимедийные объекты в презентациях

Мультимедийные объекты — это различные виды контента, которые делают презентацию более живой и наглядной:

Изображения

Звуковые и видеофайлы

Анимация

Гиперссылки и кнопки управления

2. Интерактивность в презентации

Интерактивность — это возможность взаимодействия пользователя с содержимым слайдов: нажимать кнопки, переходить по ссылкам или воспроизводить медиа. Это повышает вовлечённость аудитории.

3. Основные виды интерактивных элементов в PowerPoint:

Гиперссылки: переходы на другие слайды, сайты или файлы.

Действия: запуск видео, переход по ссылкам, запуск программ.

Кнопки и формы: использующие действия при клике.

Анимация триггеров: запуск анимации при нажатии на объект.

Вставка мультимедиа: аудио и видеофайлы, которые воспроизводятся по клику или автоматически.

Инструменты PowerPoint для работы с интерактивностью:

Вкладка Вставка: добавление видео, аудио, гиперссылок, кнопок.

Вкладка Анимация: настройка анимаций и триггеров.

Контекстное меню объектов: настройка действий.

Панель Настройка анимации: точная настройка порядка и условий запуска анимации.

Практическая часть

Задание 1. Создание интерактивного слайда с гиперссылками

1. Создайте новый слайд

Откройте PowerPoint, выберите вкладку Главная → Создать слайд → Пустой (или любой другой подходящий макет).

2. Добавьте текст или изображение

Вставьте текст: выберите Вставка → Надпись, кликните на слайде и введите текст, например, «Перейти на следующий слайд» или «Официальный сайт».

Вставьте изображение: Вставка → Рисунок, выберите изображение и разместите на слайде.

3. Выделите текст или изображение, на который хотите добавить гиперссылку.
4. Добавьте гиперссылку

Перейдите на вкладку Вставка → Гиперссылка (или нажмите Ctrl+K).

5. Настройте параметры гиперссылки:

Для перехода на слайд в текущей презентации: в разделе «Место в документе» выберите нужный слайд (например, «Слайд 3»)

Для перехода на веб-сайт: в поле «Адрес» введите URL, например, http://www.microsoft.com.

6. Подтвердите нажатием кнопки ОК.
7. Проверьте работу гиперссылки

Запустите показ презентации (клавиша F5 или вкладка Слайд-шоу → С начала)

Наведите курсор на ссылку и кликните — должен произойти переход.

Задание 2. Вставка мультимедийных объектов

Видео:

1. Перейдите на вкладку Вставка → Видео → Видео на моем компьютере.
2. Найдите и выберите нужный видеофайл (например, MP4).
3. Видео вставится на слайд — перетащите и измените размер, если нужно.
4. Определите способ воспроизведения:

Перейдите во вкладку Воспроизведение (появляется при выделении видео).

Выберите «Воспроизводить автоматически» или «Воспроизводить при клике».

Аудио:

1. Перейдите на вкладку Вставка → Аудио → Аудио на моем компьютере.
2. Выберите звуковой файл (MP3, WAV и др.).
3. На слайд добавится иконка аудио. Её можно переместить или скрыть.
4. Вкладка Воспроизведение позволяет:

- Выбрать автоматическое воспроизведение или по клику.
- Настроить зацикливание, остановку при смене слайда.

5. Для управления аудио можно добавить кнопку — например, фигуру, которой присвоить действие запуска звука (об этом в задании 3).

Задание 3. Создание интерактивных кнопок с действиями

1. Перейдите на вкладку Вставка → Фигуры.
2. Выберите форму кнопки (например, закруглённый прямоугольник).
3. Нарисуйте кнопку на слайде — кликните и протяните мышью.
4. Кликните по кнопке правой кнопкой мыши → выберите Изменить текст и

введите название (например, «Запустить видео»).

5. Чтобы задать действие при клике, правый клик → Действие.
6. В открывшемся окне «Параметры действия» перейдите на вкладку Щелчок мышью.
7. Выберите действие, например:

– «Перейти к слайду» — укажите номер слайда, на который кнопка переходит.

– «Воспроизвести звук» — выберите файл аудио или «По умолчанию» (если звук встроен).

– «Выполнить гиперссылку» — введите адрес сайта.

8. Нажмите ОК, чтобы сохранить настройку.
9. Запустите презентацию и проверьте работу кнопки.

Задание 4. Добавление анимации с триггерами

1. Выберите объект на слайде (например, картинку или фигуру), к которому хотите применить анимацию.
2. Перейдите во вкладку Анимация.
3. Нажмите Добавить анимацию (не просто выбрать, а именно добавить).
4. Выберите эффект анимации (например, «Появление», «Выцветание»).
5. Откройте панель Область анимации (вкладка Анимация → Область анимации).
6. В списке анимаций выберите нужную.
7. Нажмите правой кнопкой по выбранной анимации и выберите Триггер → Начать эффект при щелчке по.
8. В появившемся списке объектов на слайде укажите объект (например, кнопку), по клику на которую должна начаться анимация.
9. Проверьте презентацию в режиме слайд-шоу: клик по кнопке должен запускать анимацию объекта.

Задание 5. Создайте еще одну презентацию используя информацию и материалы, представленные в таблице

№ слайда	*Разметка слайда*	*Содержимое слайда (текст, рисунки, диаграммы и т.д.)*
1	Титульный слайд	«Обитатели Тайги»
2	Только заголовок	Тайга расстилается по просторам России с запада на восток. Лето в тайге не жаркое, но и не особо холодное, правда короткое, а зима длится долго с обильными снегопадами и долгими морозами. 
3	Заголовок и текст	**Заголовок:** Обитатели тайги: **Текст:**

		• сибирский бурундук; • кедровка; • рысь; • филин; • беркут; • бурый медведь; • свиристель; • глухарь.
4	Заголовок и текст	**Заголовок:** Сибирский бурундук **Текст:** Любимое лакомство - кедровые шишки. Бурундук заселяет пустые пни и дупла, неглубокие норки под корнями деревьев. А как похолодает, впадает в спячку на долгих семь месяцев! Весной зверек вылезает погреться на ярком солнышке. В это время как нельзя кстати пригодятся его припасы! Когда становится совсем тепло, самка приносит от четырех до шести бурундучат! Они растут очень быстро и через месяц навсегда покидают родительский дом.
5	Пусто слайд	
6	Заголовок и текст	**Заголовок:** Кедровка **Текст:** Эта любительница кедровых орешков! Крепким длинным клювом она ловко достает семена из спелых шишек. Потом несет корм в гнездо птенцам, свитое из веток и мха на высокой сосне. К зиме она делает запасы орехов, пряча их в мох или щели гнилых деревьев. Часть семян остается во мху и к весне прорастает: так кедровка участвует в расселении сибирской кедровой сосны. Когда орехи еще не созрели, птицы едят жуков, гусениц, семена ели, ягоды.

7	Пусто слайд	
8	Заголовок и текст	**Заголовок:** Рысь **Текст:** Грациозная лесная хищница рысь – мастер маскировки. Ее дымчато-желтую шубку трудно заметить в зарослях, будь то зима или лето. Крадется она бесшумно, будто скользит по земле. Притаится у заячьей тропы или у водопоя и терпеливо поджидает жертву. Зазевался заяц-беляк, хрустя корой осины, и не заметил, как оказался в когтях у лесной охотницы. А зимой добычей хищницы может стать и косуля, провалившаяся в снег.
9	Заголовок и текст в две колонки	**Заголовок удалить.** **1 объект:** У рыси лапы широкие, покрытые густой шерстью. Мех у рыси такой густой и теплый, что она спокойно спит на снегу. Рысята очень похожи на домашних котят, только хвостики у них короткие, лапы длинные, а на ушах кисточки. **2 объект:** 
10	Заголовок и текст	**Заголовок:** Филин **Текст:** Ни птичка, ни мышка не ускользнут от его зорких глаз и тонкого слуха. Куропатку и глухаря одолеет, а про ежа и говорить нечего. Но уж если днем пронырливые сойки и сороки обнаружат в ветвях дерева отдыхающего филина, ему не поздоровится. Птицы поднимут гвалт на весь лес! И каждая постарается клюнуть ночного разбойника. А он только топорщит перья да забирается поглубже в крону дерева.
11	Заголовок и текст в две	**Заголовок удалить.** **1 объект:**

	колонки	 **2 объект:** Зимой филин делает запасы: прячет в дупло или в снег свою добычу – мышей и птиц, - пригодятся в лютую стужу. А по весне в гнезде появляются птенцы. Родители вместе выхаживают их и кормят даже тогда, когда дети начинают летать.
12	Заголовок и текст	**Заголовок:** Беркут **Текст:** Самый известный из орлов – символ силы и мужества, - этот крупный хищник обладает крепким загнутым клювом, зорким взглядом, большими широкими крыльями и могучими лапами с острыми крепкими когтями. Паря высоко в небе, он видит на земле даже мышь. Но его желанная добыча – сурки, суслики и зайцы. Заметив зверька, беркут пикирует и в последний момент выбрасывает вперед свое оружие – лапы с мощными когтями, а через мгновение взлетает уже с добычей.
13	Пусто слайд	
14	Заголовок и текст	**Заголовок:** Бурый медведь **Текст:** Медведь – зверь всеядный. Ест ягоды, орехи и сочные корешки трав, ловит рыбу, лягушек, ящериц, мышей, птиц, поедает и их яйца, очень любит мед, личинки насекомых, в том числе муравьев, и даже ест падаль. Кормится он в основном в сумерках и ночью. На вид он тяжел и неуклюж, но бегает резво, отлично плавает и лазает по деревьям. Зимой медведи спят в берлогах под защитой бурелома, в глухих дремучих местах.

15	Пусто слайд	
16	Заголовок и текст	**Заголовок:** Свиристель **Текст:** «Свир-р-ри, свир-р-ри» - из-за звонкой трели, похожей на верещание кузнечиков, эту птичку назвали свиристелем. В мае на северной границе тайги, обычно на высоких елях, свиристели вьют гнезда из тонких веточек, мха и лишайника. Они едят насекомых, ими же кормят птенцов. Даже комаров, сбив в комочек и смочив слюной, несут в гнездо. Поздней осенью птицы собираются в стаи и откочевывают к югу в поисках пищи. Навещают сады, парки, скверы, посадки плодовых деревьев и кустарников у жилья.
17	Заголовок и текст в две колонки	**Заголовок удалить.** **1 объект:** Нередко зимой можно увидеть, как стайки хохлатых птиц облепили березу и снуют туда-сюда, издавая тонкие хрустальные трели. На березах у них столовая, а летают они за ягодами рябины, растущей поблизости. Сорвут ягоду - и обратно на березу. **2 объект:** 
18	Заголовок и текст	**Заголовок:** Глухарь **Текст:** Глухарь – самый крупный из диких родственников кур. Весной, еще до восхода солнца, самец шумно взлетает на сосну, расправляет крылья, распускает веером хвост и заводит песню: «Скрик, скрик. Тэк-тэк-тэк». Он так увлеченно и громко поет, что на несколько мгновений теряет слух. За это его и прозвали глухарем. На его призыв слетаются глухари и глухарки.

19	Пусто слайд	

3. Создайте гиперссылки по следующей схеме: на Слайде №3:

при нажатии на слово «сибирский бурундук» осуществляется переход на Слайд №4;

при нажатии на слово «кедровка» осуществляется переход на Слайд №6;

при нажатии на слово «рысь» осуществляется переход на Слайд №8;

при нажатии на слово «филин» осуществляется переход на Слайд №10;

при нажатии на слово «беркут» осуществляется переход на Слайд №12;

при нажатии на слово «бурый медведь» осуществляется переход на Слайд №14;

при нажатии на слово «свиристель» осуществляется переход на Слайд №16;

при нажатии на слово «глухарь» осуществляется переход на Слайд №18.

4. Создайте управляющие кнопки *Назад*, *Далее* и *Домой* (пункт меню Показ слайдов/Управляющие кнопки) по следующей схеме:

4.1. кнопку *Назад* разместите на Слайдах №№ 5, 7, 9, 11, 13, 15, 17, 19 (данная кнопка должна возвращать на Слайд №3);

4.2. кнопку *Далее* разместите на Слайдах №№ 4, 6, 8, 10, 12, 14, 16, 18 (она должна перемещать на следующий слайд, т.е. на Слайды №№ 5, 7, 9, 11, 13, 15, 17, 19 соответственно);

4.3. кнопку *Домой* разместите со 2-го по 19-ый слайды (она должна возвращать на 1-ый слайд).

5. На 1 слайде разместите кнопку *Выход*.

6. Оформите дизайн презентации самостоятельно.

7. Оформите эффекты анимации самостоятельно.

Контрольные вопросы:

1Для чего предназначена программа MS Power Point ?

2Из каких действий состоит процесс создания презентаций?

3Что такое слайд?

4Как добавить в презентацию новый слайд?

5Что такое шаблон?

6 Как удалить слайд?

7Как изменить порядок слайдов в презентации?

8 Как изменить фон и цвета на слайде?

9 Какие существуют режимы просмотра презентации?

10 Как включить режим полноэкранного просмотра презентации?
11 Как добавить на слайд диаграмму
12 Как добавить на слайд таблицу?
13 Как добавить на слайд текстовую надпись?
14 Как изменить маркировку пунктов списка на слайде?
15 Как изменить шрифт для текста на слайде?
16 Как изменить положение текстовой надписи на слайде?
17 Для чего нужен режим «Сортировщик слайдов»?
18 Как создаются управляющие кнопки? Для чего их можно использовать?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 14

Тема: Создание веб-страницы включающей текст, мультимедийные объекты (рисунки, звуковые данные, видео). Создание форм на HTML.

Количество часов на выполнение: 2.

Цель работы: Создать веб-страницу на HTML, которая содержит:

текстовый контент;

изображения (рисунки);

звуковые файлы (аудио);

видеофайлы;

форму для ввода данных пользователем (текстовое поле, кнопки и др.).

Оборудование: Персональный компьютер

Методика выполнения задания
Теоретическая часть

Основы HTML

HTML (HyperText Markup Language) — язык разметки для создания веб-страниц. Он описывает структуру и содержимое страницы с помощью тегов.

Основная структура HTML-документа:

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Заголовок страницы</title>
</head>
<body>
  <!-- Контент страницы -->
</body>
</html>
```

<!DOCTYPE html> — объявление типа документа.
<html> — корневой тег.
<head> — блок с информацией о странице (метаданные).
<body> — содержимое видимое пользователю.

Текстовые элементы
Заголовки: <h1>, <h2>, ..., <h6> — используются для заголовков разного уровня важности.
Абзацы: <p> — для текста.
Форматирование текста: , ,
 (перенос строки) и др.
Работа с мультимедиа - HTML5 расширил возможности встраивания мультимедийного контента без плагинов.
Изображения - Тег <img> вставляет картинку:

```html
<img src="filename.jpg" alt="Описание" width="300" />
```

src — путь к файлу изображения.
alt — альтернативный текст, показывается, если изображение недоступно.
Можно задавать размеры через width и height.

Аудио
Тег <audio> предназначен для встраивания звуковых файлов:

```html
<audio controls>
  <source src="sound.mp3" type="audio/mpeg" />
  Ваш браузер не поддерживает элемент audio.
</audio>
```

Атрибут controls добавляет стандартные кнопки управления.
Внутри можно указать несколько источников с разными форматами для совместимости.

Видео

Тег <video> встраивает видеофайлы:

```html
<video width="640" height="360" controls>
  <source src="video.mp4" type="video/mp4" />
  Ваш браузер не поддерживает элемент video.
</video>
```

Также поддерживается управление воспроизведением через controls.
Можно указать размеры видео.

Создание форм

Формы позволяют пользователям вводить и отправлять данные на сервер.

Основные теги и атрибуты

Тег <form> определяется форма.

```html
<form action="url" method="post">
  <!-- поля формы -->
</form>
```

action — адрес, куда отправляются данные;
method — способ отправки (get или post).

Поля ввода

<input> — универсальный тег для разных типов ввода, в зависимости от атрибута type.

type="text" — однострочное текстовое поле.
type="email" — поле для email.
type="password" — поле для пароля.
type="submit" — кнопка отправки.
<textarea> — многострочное поле для текста.
<label> — подпись для поля, улучшает доступность.

Обязательные поля

Атрибут required заставляет пользователя заполнить поле перед отправкой.

Практическая часть

Задание 1.

Шаг 1. Создание базового HTML-файла

Откройте любой текстовый редактор (например, Visual Studio Code, Notepad++, Sublime Text или даже обычный Блокнот).

Создайте новый файл и сохраните его с расширением .html, например, index.html.

Впишите базовую структуру HTML-документа:

```html
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8" />
  <title>Моя мультимедийная страница</title>
</head>
<body>

</body>
</html>
```

Задание 2. Добавление текстового контента
В теле документа <body> добавьте заголовок и абзац:

```html
<h1>Добро пожаловать на мою веб-страницу</h1>
<p>Здесь вы найдете разные мультимедийные объекты и форму для обратной связи.</p>
```

Задание 3. Вставка изображения (рисунка)
Подготовьте файл изображения (например, picture.jpg) и положите его в ту же папку, где находится ваш HTML-файл.
Добавьте следующий код под текстом:

```html
<h2>Изображение</h2>
<img src="picture.jpg" alt="Пример изображения" width="400" />
```

- Атрибут alt — альтернативный текст на случай, если изображение не загрузится.
- Ширина 400 пикселей для удобного отображения (можно изменить).

Задание 4. Вставка аудио (звукового файла)
Подготовьте аудио файл (например, sound.mp3) и сохраните его в той же папке.
Добавьте следующий код:

```html
<h2>Аудио</h2>
<audio controls>
  <source src="sound.mp3" type="audio/mpeg" />
  Ваш браузер не поддерживает элемент audio.
</audio>
```

- controls — отображает контролы плеера (воспроизведение, пауза и т. д.).
- Внутри тега <audio> можно указать несколько источников, если нужно.

Задание 5. Вставка видео
Добавьте видеофайл video.mp4 в папку.
Включите видео с контролами:

```html
<h2>Видео</h2>
<video width="560" height="315" controls>
  <source src="video.mp4" type="video/mp4" />
  Ваш браузер не поддерживает элемент video.
</video>
```

- Ширина 560, высота 315 — стандартный размер (можно менять).
- Контролы для управления воспроизведением.

Задание 6. Создание формы
Теперь создадим простую форму обратной связи с полями:

Имя (текстовое поле)
E-mail (текстовое поле)
Сообщение (абзац для ввода текста)
Кнопка отправки

```html
<h2>Свяжитесь с нами</h2>
<form action="#" method="post">
  <label for="name">Имя:</label><br />
  <input type="text" id="name" name="name" placeholder="Введите ваше имя" required /><br /><br />

  <label for="email">E-mail:</label><br />
  <input type="email" id="email" name="email" placeholder="Введите ваш email" required /><br /><br /

  <label for="message">Сообщение:</label><br />
  <textarea id="message" name="message" rows="5" cols="40" placeholder="Ваше сообщение"></textarea><

  <button type="submit">Отправить</button>
</form>
```

• Атрибут action="#" указывает, куда отправлять данные (здесь - никто не обрабатывает, для обучения достаточно).

• Атрибут required задает обязательные поля.

• label улучшает доступность.

Задание 7. Сделать скриншот - Общий итоговый код веб-страницы

Задание 8. Запуск и проверка страницы

Откройте папку с сохранённым файлом.

Дважды кликните по файлу index.html — он откроется в браузере.

Проверьте отображение текста, изображения, аудио, видео.

Проверьте работу формы: при нажатии кнопки «Отправить» страница обновится, т.к. action="#".

Контрольные вопросы:

1. Что обозначает тег <img>? Какие обязательные и рекомендуемые атрибуты у него есть?
2. Чем отличается атрибут src от атрибута alt в теге изображения?
3. Для чего используется тег <audio> и как включить элементы управления аудио?
4. Какие форматы видео можно встроить с помощью тега <video>?
5. Что такое атрибут action в теге <form> и зачем он нужен?
6. Какие типы <input> встречаются чаще всего при создании форм?
7. Как обозначить обязательное поле в форме?
8. Чем отличается метод отправки формы GET от POST?
9. Почему важно использовать тег <label> для полей формы?
10. Как сделать так, чтобы пользователю был показан текст, если браузер не поддерживает вставленный мультимедиа элемент?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 15

Тема: Оформление страницы с помощью каскадных таблиц стилей.

Количество часов на выполнение: 2.

Цель работы: Освоить навыки оформления веб-страницы с использованием CSS, научиться подключать внешние таблицы стилей, применять различные CSS-

свойства для улучшения внешнего вида и структуры сайта, а также понять принципы организации и каскадности стилей для создания привлекательного и удобочитаемого веб-интерфейса.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

CSS (Cascading Style Sheets — каскадные таблицы стилей) предназначены для оформления веб-страниц: установки цветов, шрифтов, размеров, расположения элементов и т. д. CSS позволяет отделить содержимое страницы от её оформления, что облегчает сопровождение и развитие сайта.

1. Основные понятия CSS

Селекторы — определяют, к каким элементам применяется стиль.

Свойства — атрибуты, которые настраиваются (цвет, размер, шрифт и т. д.).

Значения — конкретные параметры свойства (например, red, 16px, Arial).

Пример:

```
p {
  color: red;
  font-size: 16px;
}
```

Это стиль для всех абзацев <p>.

2. Варианты подключения CSS

2.1 Встроенный стиль (inline)

Добавляется прямо в тег через атрибут style:

```
<p style="color: blue;">Текст</p>
```

2.2 Внутренний стиль (в разделе <style>)

Добавляется внутри <head> документа:

```
<head>
  <style>
    p {
      color: green;
    }
  </style>
</head>
```

2.3 Внешний файл CSS

Наиболее распространённый способ:

Создать файл styles.css.

Подключить его в HTML:

```
<head>
  <link rel="stylesheet" href="styles.css" />
</head>
```

Основные свойства CSS и их применение

Свойство	Назначение	Пример

Свойство	Назначение	Пример
color	Цвет текста	color: red;
background-color	Фон страницы или элемента	background-color: #f0f0f0;
font-family	Шрифт текста	font-family: Arial, sans-serif;
font-size	Размер текста	font-size: 18px;
margin	Внешние отступы	margin: 10px;
padding	Внутренние отступы	padding: 5px;
border	Граница элемента	border: 1px solid black;
width и height	Размеры элементов	width: 200px; height: 100px;
text-align	Выравнивание текста	text-align: center;

Каскадность и специфичность

Каскадность означает, что при конфликте нескольких стилей применяется тот, который важнее по приоритету.

Специфичность зависит от селектора (например, более конкретный селектор переопределяет менее конкретный).

Типы селекторов

Теговые: p, h1, div

Классовые: .classname

Идентификаторные: #idname

Группировка: h1, h2, h3

Комбинированные: div p, .menu ul

Псевдоклассы: a:hover, p:first-child

Основные правила оформления страницы

Используйте внешние таблицы стилей для удобства.

Назначайте классы (class) для группировки элементов.

Используйте комментарии (/* комментарий */) для документации стилей.

Следите за каскадностью и специфичностью при написании правил.

Практическая часть:

Задача

Создайте простую HTML-страницу, оформленную с помощью CSS, которая включает:

Заголовок (например, <h1>)

Несколько абзацев

Секцию с фоном и рамкой

Кнопку, стилизованную с помощью CSS

Используйте внешнюю таблицу стилей style.css

Шаги выполнения:

Создайте файл index.html и подключите к нему внешний файл CSS.

В HTML-оформлении используйте теги:

```
<!DOCTYPE html>
<html lang="ru">
<head>
<meta charset="UTF-8" />
<title>Практическая работа по CSS</title>
<link rel="stylesheet" href="style.css" />
```

```
</head>
<body>
<h1>Добро пожаловать!</h1>
<section class="intro">
  <p>Это пример страницы, оформленной с помощью CSS.</p>
  <p>Цвет, шрифты, отступы — всё настроено через CSS.</p>
</section>
<div class="button-container">
  <button class="styled-button">Кнопка</button>
</div>
</body>
</html>
```

В файле style.css задайте стили для элементов:

```
/* Общие настройки страницы */
body {
  font-family: Arial, sans-serif;
  background-color: #f9f9f9;
  margin: 20px;
}

/* Заголовок */
h1 {
  color: #336699;
  text-align: center;
}

/* Секция с классом intro */
.intro {
  background-color: #ffffff;
  border: 2px solid #ccc;
  padding: 20px;
  margin-bottom: 20px;
}

/* Абзацы внутри секции */
.intro p {
  font-size: 16px;
  line-height: 1.5;
}

/* Стиль для кнопки */
.styled-button {
  background-color: #4CAF50;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
```

```
}

.styled-button:hover {
  background-color: #45a049;
}
```

Проверьте работу страницы в браузере: убедитесь, что стили применились правильно и внешний вид соответствует заданию.

Контрольные вопросы

1. Что такое CSS и зачем он нужен при оформлении веб-страниц?
2. Как подключить внешний файл CSS к HTML-документу?
3. Какие виды селекторов существуют в CSS? Назовите основные.
4. Чем отличаются селекторы по тегу, классу и id?
5. Какие свойства CSS отвечают за фон страницы и цвет текста?
6. Как задать внутренние и внешние отступы у элементов?
7. Что такое каскадность и приоритетность правил CSS?
8. Как сделать кнопку, которая меняет цвет при наведении мышью?
9. Почему лучше использовать внешний CSS, а не встроенные стили?
10. Что такое псевдоклассы в CSS? Приведите пример.
11. Как можно группировать несколько элементов один раз?
12. Как обеспечить читаемость и удобство сопровождения CSS-кода?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 16

Тема: Исследование физической модели

Количество часов на выполнение: 4

Цель работы: Смоделировать равномерное движением тела.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Физические модели — это упрощённые, абстрактные представления реальных физических систем, созданные для упрощения исследования их поведения, объяснения явлений и проведения экспериментов без необходимости установки сложных и дорогих физических приборов.

Цели и задачи моделирования

Позволяет понять основные свойства и законы системы.

Облегчает проведение экспериментов.

Позволяет делать предсказания о поведении системы при различных условиях.

Основные типы физических моделей

Деформационные модели: моделируют изменение формы объектов при воздействии сил.

Гидродинамические модели: изучают движение жидкости и газа.

Механические модели: реализуют динамику движений тел и систем.

Тепловые модели: изучают теплопередачу, расширение тел и другие процессы.

Важные понятия

Модель — упрощённое представление системы.

Обоснованность— насколько модель отражает основные свойства реальной системы.

Точность — степень соответствия модели реальности.

Практическая часть

Задание 1: Создание простейшей физической модели

Смоделировать равномерное движением тела.

Задача: Представьте, что вы моделируете свободное падение небольшого объекта в вакууме. Опишите, какие параметры вам понадобятся, чтобы учесть влияние силы тяжести и сопротивления воздуха в вашей модели.

Развернутый ответ:

Основные параметры: масса объекта, ускорение свободного падения, коэффициент сопротивления воздуха, начальная скорость, время падения.

В модели необходимо учесть силу тяжести ($F=mg$) и силу сопротивления воздуха, пропорциональную скорости ($F_{сопр} = kv$).

Полученная модель можно описывать дифференциальными уравнениями.

Задание 2: Моделирование поведения углового маятника

Задача: В условиях малых колебаний сформулируйте уравнение движения для математического маятника и опишите, как моделировать амплитуду и частоту колебаний.

Задание 3: Анализ достоверности модели

Цель: Оценить, насколько выбранная модель подходит для описания реальной ситуации.

Задача: Для модели маятника обсудите, при каких условиях она максимально приближена к реальности и какие факторы при этом можно упустить.

Решение:

Модель хороша при малых углах отклонения ($\theta \ll 1$).

Можно упустить сопротивление воздуха, неровности нити, растяжимость нити, наличие трения и др.

В реальности амплитуда колебаний со временем уменьшается под действием сил трения и сопротивления

.

Контрольные вопросы:

1. Что такое физическая модель и зачем она нужна?
2. Назовите основные типы физических моделей.

3. Какие параметры необходимо учесть при моделировании свободного падения тела в воздухе?
4. Как формулируется уравнение движения для математического маятника?
5. Почему в моделях иногда упускают сопротивление воздуха или трение?
6. Чем отличается идеальная модель от реальной?
7. Какие преимущества даёт использование физических моделей?
8. В каких случаях модель может оказаться неиспользуемой или плохой?
9. Объясните, как можно проверить точность модели?
10. Какое значение имеет упрощение модели для практических расчетов?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 17

Тема: Средства искусственного интеллекта

Количество часов на выполнение: 4

Цель работы:

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Машинный перевод (Machine Translation, MT) — это технология связного перевода текстов компьютерной программой с одного естественного языка на другой.

Машинный перевод сегодня является привычным инструментом. Но так было не всегда. Идея использования машинной памяти для создания словарей принадлежит Чарльзу Бэббиджу, создавшему аналитическую машину. Идея Ч. Бэббиджа заключалась в том, что память вычислительной машины можно использовать для хранения словарей. Однако воплощение этой идеи случилось лишь спустя почти 100 лет.

История машинного перевода практически начинается с 7 января 1954 года, когда состоялась демонстрация возможностей машинного перевода, получившая название Джорджтаунский эксперимент. Презентация была подготовлена одноименным университетом совместно с фирмой IBM. В процессе демонстрации было переведено около 60 фраз с русского языка на английский. Система содержала 250 слов и 6 грамматических правил. На следующий день результаты эксперимента были опубликованы в газете «Нью-Йорк таймс».

Примерно в то же время происходили исследования машинного перевода в СССР. В начале 1955 года Академией Наук СССР были созданы две исследовательские группы — в Математическом Институте имени В. А. Стеклова под руководством выдающегося математика и кибернетика Алексея Ляпунова и в Институте точной механики и вычислительной техники АН СССР под руководством математика Д. Ю. Панова. Результаты первых экспериментов по

машинному переводу, проведенных на компьютере БЭСМ, Д. Ю. Панов опубликовал же в 1956 году.

Первоначально в системах машинного перевода были заложены алгоритмы последовательного перевода «слово за словом», «фраза за фразой». При таком подходе возможности систем находились в прямой зависимости от памяти компьютера. Перевод текста осуществлялся предложение за предложением, а смысловые связи между ними никак не учитывались. Такие системы называют системами прямого перевода.

Первые коммерческие системы машинного перевода появились в середине 80-х годов прошлого столетия. Они пользовались большой популярностью, были реализованы на персональных компьютерах, поэтому имели значительно большие объёмы словарей, чем системы первого поколения. Однако, эти системы всё ещё не умели анализировать и синтезировать тексты. Они оставались системами прямого перевода.

В 1993 г. под руководством профессора Г. Г. Белоногова (ВИНИТИ) была создана промышленная версия системы RETRANS фразеологического машинного перевода с русского языка на английский и обратно, которая применялась в министерствах обороны, путей сообщения, науки и технологий, а также во ВНТИЦ.

В начале 90-х стали создаваться отечественные фирмы, производящие коммерческие продукты машинного перевода, такие как «Виста Текнолоджиз» и «Адвентис», ПРОМТ, «Медиа Лингва».

Глобализация современного общества приводит к тому, что люди из разных стран обмениваются документами на разных языках. Перевод вручную требует достаточного количества времени. Для ускорения процесса используются системы компьютерного перевода текста. К преимуществам систем машинного перевода можно отнести следующие:

Высокая скорость перевода, в связи с значительным сокращением времени, требуемого для перевода текстов.

Низкая стоимость перевода. Часто при переводе нужно уловить только смысл письма или страницы в Интернете, а профессиональные переводчики требуют оплаты всех страниц текста.

Конфиденциальность. Перевод личных писем, финансовых документов и др. не всегда можно доверить постороннему лицу.

Универсальность. При правильных настройках программа-переводчик справится с переводом текстов из самых разных областей, а у профессионального переводчика всегда есть своя специализация.

Перевод в режиме онлайн и перевод содержания Интернет-страниц. Сервисы онлайн-перевода всегда под рукой и помогут в нужный момент быстро перевести информацию, даже без программы-переводчика.

На сегодняшний день разработано большое количество программ, помогающих автоматизировать перевод текста. Их можно разделить на две большие группы — компьютерные словари и системы компьютерного перевода текста.

У компьютерных словарей можно выделить такие свойства, как:

— Многоязычность, т. е. выбор языков и направления перевода.

— Специализация, когда в дополнение к основному словарю могут содержать словари по областям знаний (биоинформатика, география и т. д.).

— «Быстрый набор», когда в процессе набора слова возникает список похожих слов, возможность работы с словосочетаниями.

— Мультимедийность, например, прослушивание слов в исполнении диктора.

— Онлайн доступ, компьютерные словари с онлайн доступом позволяют выбрать тематический словарь и направление перевода.

Среди современных подходов к реализации алгоритмов машинного перевода (МП) выделяют два основных типа:

Классический (rule-based machine translation — RBMT), который основан на лингвистической информации об исходном и переводном языках. Состоит из двуязычных словарей и грамматик, охватывающих основные закономерности каждого языка.

Статистический (Statistical machine translation — SMT), который основан на анализе массивов текстов, представленных одновременно на языке оригинала и языке перевода.

Производители систем МП разрабатывают и применяют гибридные системы, использующие преимущества указанных типов МП.

Сегодня существует достаточно много компаний, разрабатывающих программы машинного перевода, но на мировом рынке лидируют продукты двух организаций — зарубежная компания Systran и российская компания ПРОМТ. К другим крупным производителям относятся Linguatec и Langenscheidt (Германия), Transparent Language, Babylon, Translation Experts, японо-тайская компания Asia Online и др.

Многие производители систем машинного перевода в качестве рекламы предлагают онлайн версии своих продуктов. Перечислим лишь некоторые системы онлайн перевода текста с производителями:

Translate.ru (онлайн-переводчик компании ПРОМТ)
SYSTRANet (Systran)
Google Translate (Google)
Free Translation (SDL)
Babel Fish (Systran)
Worldlingo (Systran)
InterTran (Translation Experts Limited)
ImTranslator (Smartlink Corp)
Windows Life Translator (Microsoft)
Яндекс Переводчик (Яндекс)
ABBYY Lingvo для Windows 8 Touch (ABBYY)

Развитие компьютерных технологий привело к тому, что многие функции, которые ранее мог выполнять только человек, теперь передаются на выполнение тем или иным сервисам или устройствам. За последние несколько лет голосовые помощники плотно вошли в нашу жизнь, а для кого-то они стали просто незаменимы. Алиса от Яндекса, Siri от Apple, Google Assistant от одноименной компании, Cortana от Microsoft, Echo от Amazon и др. — это программы, и для их разработки используется распознавание устной речи. А если Вы умеете программировать, то такого голосового помощника, а может быть и двоих, сможете написать для себя самостоятельно. Например, так, ка это сделал Григорий Бакунов из Яндекса.

Всё рассмотренное ранее имеет непосредственное отношение к искусственному интеллекту (ИИ).

Искусственный интеллект — это наука и технология, включающая набор средств, позволяющих компьютеру на основании накопленных знаний давать ответы на вопросы и делать на базе этого экспертные выводы, т. е. получать знания, которые в него не закладывались разработчиками.

Наука под названием «искусственный интеллект» относится к компьютерным наукам. Значит, системы ИИ — это компьютерные системы, использующие технологии ИИ.

Принято разделять системы ИИ на слабый (прикладной) ИИ и сильный (общий) ИИ.

Сильный или универсальный ИИ способен выполнить любую человеческую задачу.

Слабый ИИ предназначен для узкого спектра задач. Такие системы могут делать только одно дело. Все рассмотренные выше примеры относятся к слабому или прикладному ИИ. Искусственный интеллект активно развивается и имеет огромную популярность в последнее время. Ежедневно сообщается о новых системах ИИ, решающих прикладные задачи в разных областях науки, техники, медицины и др.

Для представления разнообразия решаемых задач, приведём лишь некоторые цитаты новостных лент:

— «Инженеры из Массачусетского технологического института (MIT) разработали автономных роботов размером с клетку человека. Устройства будут использоваться для исследования организма, а также диагностики газо- и нефтепроводов. Об этом пишет The Verge».

— «Ученые автоматизировали поиск белковых кристаллов с помощью компьютерного зрения».

— «В Стэнфорде создали нейросеть Decagon, прогнозирующую побочные эффекты лекарств».

— «Группа инженеров из Сколтеха создала искусственный интеллект, который поможет ученым подбирать самые быстрорастущие и выносливые растения для будущих космических экспедиций».

— «Инженеры из Университета Южной Австралии, Штутгартского университета, Университета Флиндерса и Института имени Макса Планка в Германии научили искусственный интеллект анализировать характер человека по его глазам».

— «Исследователи из лаборатории армии США и Института робототехники университета Карнеги — Меллона предложили метод быстрого обучения роботов для автономной работы при минимальном контроле человека».

Отдельное большое направление ИИ занимает робототехника. Примеров использования ИИ в робототехнике можно приводить очень много уже сегодня. Существуют фирмы, такие как Boston Dynamics, которые давно и успешно разрабатывают роботов, обучают их, проводят испытания. Компания Kuka занимается выпуском роботов для производств. На больших складах крупных компаний используют роботов-доставщиков. Развивающимся сегодня и очень перспективным направлением считаются беспилотные летательные аппараты.

Многие из Вас наверняка слышали о системах распознавания лиц. Здесь тоже не обошлось без ИИ. Идентификация и поиск изображений — ещё одно из направлений ИИ. Одна из систем идентификации представлена в павильоне «Умный город» на ВДНХ в Москве. Система определяет пол и возраст по записи с видеокамеры. Множество других интеллектуальных систем, используемых в городе, представлено в павильоне. Они касаются образования, медицины, строительства и др.

Вернёмся к Алисе, с которой начинался урок. Она не только расскажет о погоде и поможет вызвать такси. Посмотрим на навыки Алисы — она поиграет с Вами в города, проведёт викторину по истории, устроит Тотальный диктант, прочитает стихотворения классиков и многое, многое другое. Здесь и обучающие системы, и компьютерные игры.

На сегодняшнем уроке мы познакомились с системами машинного перевода.

Среди современных подходов к реализации алгоритмов машинного перевода (МП) выделяют два основных типа — на основе правил и на основе статистики.

Узнали, что работы над интеллектуальными системами ведутся уже более полувека. Определились с тем, что такое искусственный интеллект.

Искусственный интеллект — это наука и технология, включающая набор средств, позволяющих компьютеру на основании накопленных знаний давать ответы на вопросы и делать на базе этого экспертные выводы, т. е. получать знания, которые в него не закладывались разработчиками.

Рассмотрели лишь некоторые направления искусственного интеллекта.

Практическая часть

Задание 1. Откройте браузер на странице quickdraw.withgoogle.com. Попробуйте выполнить задание – нарисовать несколько предметов, которые нейронная сеть попробует угадать. Сколько из нарисованных вами изображений нейронная сеть определила правильно? Прикрепите в отчёт снимок экрана.

Задание 2. С помощью сервиса color.artlebedev.ru раскрасьте чёрно-белое изображение (можно использовать файл boat.jpg или своё изображение) и добавьте в отчёт оригинал и полученное цветное изображение.

Оригинал	Цветной вариант

Задание 3. С помощью сервиса www.captionbot.ai, использующего нейронные сети, постройте подпись к вашему изображению и переведите её на русский язык.

Оригинал	На русском языке

Задание 4. С помощью сервиса www.how-old.net попытайтесь определить возраст человека, который изображен на фотографии (можно использовать свою фотографию или изображение girl.jpg).

Задание 5. С помощью сервиса hi.cs.waseda.ac.jp:8082 выполните раскраску чёрно-белого изображения.

Оригинал	Цветной вариант

Сравните результат с тем, который был получении в задании 2.

Задание 6. С помощью сервиса bigjpg.com/ru выполните увеличение какого-нибудь изображения в 4 раза (можно использовать файл flowers.png). Добавьте в отчёт оригинал и полученное увеличенное изображение.

Оригинал	Увеличенное изображение

Контрольные вопросы:

Что такое Искусственный интеллект?

Какие ИИ вы знаете?

В чём преимущества ИИ?

В чём недостатки ИИ?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 18

Тема: Обработка данных. Вычисление суммы, среднего арифметического, наибольшего и наименьшего значений. Сортировка, фильтрация, условное форматирование.

Количество часов на выполнение: 4ч

Цель работы: уметь применять технологию обработки информации в электронных таблицах.

Оборудование: Персональный компьютер, Microsoft Excel

Методика выполнения задания

Теоретическая часть

Прикладные программы, предназначенные для работы с данными, представленными в таблицах, называются **табличными процессорами** или **электронными таблицами**. Первый табличный процессор был

создан в 1979 году и предназначался для автоматизации рутинных вычислительных процедур. Современные электронные таблицы применяются не только для выполнения расчётов.

Наиболее распространёнными табличными процессорами являются Microsoft Excel и OpenOffice Calc.

После запуска программы Microsoft Excel на экране открываются два окна: окно табличного процессора и окно созданного в нём документа. Документ, создаваемый в табличном процессоре, называется **рабочей книгой** и по умолчанию получает имя **Книга1**. Вновь созданная рабочая книга состоит из трёх листов с именами **Лист1**, **Лист2** и **Лист3**. Имена листов указываются на ярлычках. Пользователь может переименовать листы по своему усмотрению, добавить к книге новые листы или удалить ненужные. В окне рабочей книги отображается содержимое текущего листа. Рабочая область листа с электронной таблицей **столбцами** и **строками** разбита на **ячейки**. Столбцы обозначены буквами латинского алфавита, строки пронумерованы. Адрес ячейки образуется из имени столбца и номера строки, на пересечении которых она находится.

Ячейка — это наименьшая структурная единица электронной таблицы, которая образуется на пересечении столбца и строки.

Две и более ячейки листа электронной таблицы образуют диапазон ячеек. При задании адреса связного диапазона указывают его начальную и конечную ячейки — ячейки левого верхнего и правого нижнего углов (например, A1:A10). Чтобы указать адрес несвязного диапазона ячеек, надо через точку с запятой указать адреса его связных частей.

В таблице приведены объекты табличного процессора, а также их основные свойства, которые далее будут рассмотрены более подробно.

Некоторые приёмы ввода и редактирования данных

Вся информация заносится пользователем в ячейки электронной таблицы. Для того чтобы вводить или редактировать данные в той или иной ячейке, в неё следует поместить табличный курсор, т. е. сделать ячейку активной.

Содержимым ячейки может быть число, текст или формула. Электронные таблицы работают с данными следующих типов:

— числовые значения (например, 143; 51,2; 4/5; 1,23E+02);
— дата и время суток (например, май 2018; 31.12.2000; 15:00; 3:00 PM);
— формулы (например, =(A1+B1)/2; =СУММ(A1:A5));
— текстовые значения (например: Всего; Фамилия);
— примечания;
— гиперссылки;
— различные графические изображения.

Табличный процессор самостоятельно пытается распознать тип вводимых данных. По умолчанию числа выравниваются по правому краю ячейки.

Дробную часть числа от целой отделяют запятой или точкой, в зависимости от установок операционной системы. В русскоязычных версиях Windows в качестве разделителя целой и дробной частей числа по умолчанию используется запятая, а при употреблении точки число интерпретируется как дата.

Ввод формулы начинается со знака равенства, который указывает табличному процессору на необходимость выполнения вычислений в соответствии со следующим за ним выражением. При вводе формул необходимо соблюдать следующие правила:

1. Для обозначения арифметических действий используются операторы: +, –, *, / соответственно для сложения, вычитания, умножения и деления.
2. Для обозначения действия возведения в степень используется оператор ^;

например, 5^3 будет записано как 5^3.

3. Для обозначения действия нахождение процентов используется оператор %; например, формула нахождения 25% от числа 240 будет выглядеть так: =240*25%.
4. Нельзя опускать оператор умножения.
5. Порядок выполнения операций совпадает с порядком, принятым в математике.
6. Для изменения порядка выполнения действий используют круглые скобки.
7. Формула должна быть записана линейно, т. е. в виде строки символов.

Как правило, в формулах используются не сами исходные данные, а **ссылки** на ячейки, в которых эти данные находятся. При изменении данных в каких-либо ячейках происходит автоматический пересчёт значений всех формул, содержащих ссылки на эти ячейки. Возможность **автоматического пересчёта** формул при изменении исходных данных — одна из ключевых идей электронных таблиц. Благодаря этому электронные таблицы называют динамическими.

Практическая часть

Задание 1.

В таблице приведена информация о длине некоторых рек, протекающих по территории Ивановской области, впадающих в реку Волга

Река	Длина (км)
Елнать	54
Кинешемка	34
Казоха	9
Мера	152
Шача	58
Солоница	132
Сунжа	45

1. Найдите среднюю длину рек
2. Найдите длину рек в среднем (медиану данных)
3. Найдите размах длины рек (самая длинная минус самая короткая)

Задание 2.

На соревнованиях по фигурному катанию судьи поставили спортсмену следующие оценки:

5,2 5,4 5,5 5,4 5,1 5,1 5,4 5,5 5,3

Для полученного ряда чисел найдите сумму, среднее арифметическое, наибольшее и наименьшее значения.

Задание 3.

Численность населения (млн. чел.) и площадь (тыс. км^2) для различных континентов представлена в таблице.

Континент	Численность населения	Площадь	Плотность населения
Европа	685	10532	
Африка	510	30319	
Азия	2700	44387	
Северная и Центральная Амерка	390	24249	
Южная Америка	260	17832	

Австралия	25	8510	
Все континенты			

Вычислите среднюю плотность населения (с точностью до 1 чел./км^2) для различных континентов и в целом по планете.

Отсортируйте по столбцам:

Численность населения, Плотность населения в порядке убывания.

Площадь в порядке возрастания

Контрольные вопросы

1. Что такое электронная таблица?
2. Как задается адрес ячейки, адрес диапазона ячеек?
3. С какими типами данных работает MS EXCEL?
4. Какой символ нужно нажать в MS EXCEL, чтобы начать ввод формул?
5. В чем отличие между абсолютными и относительными ссылками в MS EXCEL?
6. Как в MS EXCEL записать абсолютную ссылку на ячейку?
7. Как в MS EXCEL записать относительную ссылку на ячейку?
8. Какие типы диаграмм позволяет использовать MS EXCEL?
9. Из каких объектов состоит диаграмма в MS EXCEL?
10. Что означает сообщение об ошибке #### в ячейке MS EXCEL?
11. Как осуществить сортировку данных в MS EXCEL?
12. Для чего служит фильтр в MS EXCEL?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 19

Тема: Численное решение уравнений с помощью подбора параметра.

Количество часов на выполнение: 2

Цель работы: Овладение практическими навыками решения уравнений в среде Exel функцией подбора параметров.

Оборудование: Персональный компьютер

Методика выполнения задания
Теоретическая часть

Встроенные математические и статистические функции.

MS Excel содержит 320 встроенных функций. Простейший способ получения

полной информации о любой из них заключается в использовании меню **Справка**. Для удобства функции в Excel разбиты по категориям (математические, финансовые, статистические и т.д.). Обращение к каждой функции состоит из двух частей: имени функции и аргументов в круглых скобках.

Таблица - Встроенные функции Excel

Функции	Вид записи	Назначение
Математические	КОРЕНЬ(...)	Вычисление квадратного корня
	ABS(...)	Вычисление абсолютного значения (модуля) числа
	ЦЕЛОЕ(...)	Округление числа или результата выражения, указанного в скобках, до ближайшего меньшего (!) целого
	ПИ( ) *	Значение математической константы «ПИ» (3,1415926...)
	НОД(…)	Наибольший общий делитель нескольких чисел
	НОК(…)	Наименьшее общее кратное нескольких чисел
	СЛЧИС( ) *	Вычисление случайного числа в промежутке между 0 и 1
Статистические	МИН(...)	Определение минимального из указанных чисел
	МАКС(…)	Определение максимального из указанных чисел
	СРЕДНЕЕ(...)	Определение среднего значения указанных чисел
	СУММ(...)	Определение суммы указанных чисел
Дата и время	СЕГОДНЯ ( ) *	Значение сегодняшней даты в виде даты в числовом формате
	МЕСЯЦ(дата)	Вычисление порядкового номера месяца в году по указанной дате
	ДЕНЬ(дата)	Вычисление порядкового номера дня в месяце по указанной дате
	ГОД(дата)	Вычисление года по указанной дате
Логические	И(условие1; условие2;...)	Вычисление значения (ИСТИНА, ЛОЖЬ) логической операции И
	ИЛИ(условие 1; условие2;...)	Вычисление значения (ИСТИНА, ЛОЖЬ) логической операции ИЛИ
	ЕСЛИ(условие; знач_ИСТИНА; знач_ЛОЖЬ)	Вычисление значения в зависимости от выполнения условия

* Записывается без аргументов.

Программа Excel радует своих пользователей множеством полезных инструментов и функций. К одной из таких, несомненно, можно отнести Подбор параметра. Этот инструмент позволяет найти начальное значение исходя из конечного,

которое планируется получить. Давайте разберемся, как работать с данной функцией в Эксель.

Зачем нужна функция Подбор параметра

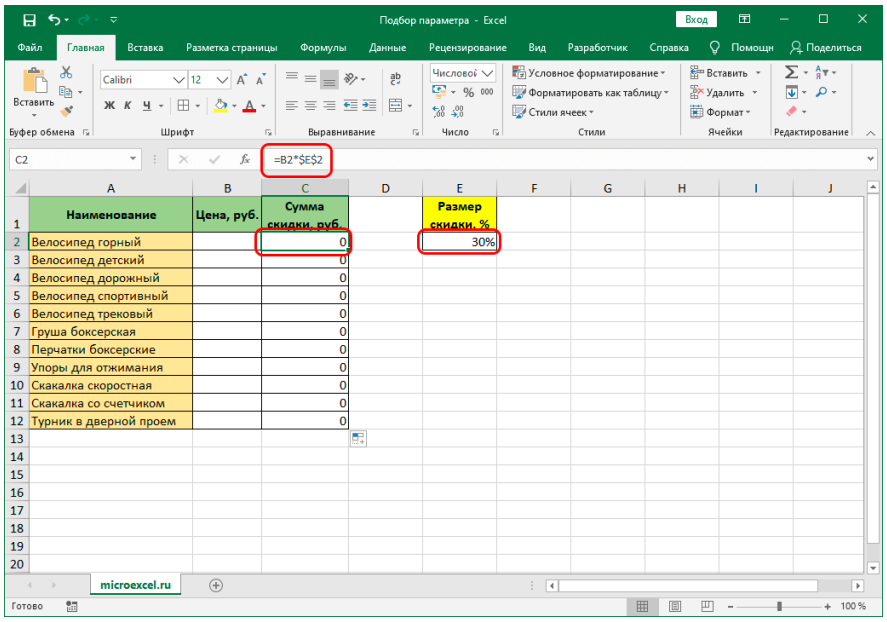
Как было уже выше упомянуто, задача функции *Подбор параметра* состоит в нахождении начального значения, из которого можно получить заданный конечный результат. В целом, эта функция похожа на *Поиск решения*, однако, при этом является более простой.

Применять функцию можно исключительно в одиночных формулах, и если потребуется выполнить вычисления в других ячейках, в них придется все действия выполнить заново. Также функционал ограничен количеством обрабатываемых данных – только одно начальное и конечное значения.

Использование функции Подбор параметра

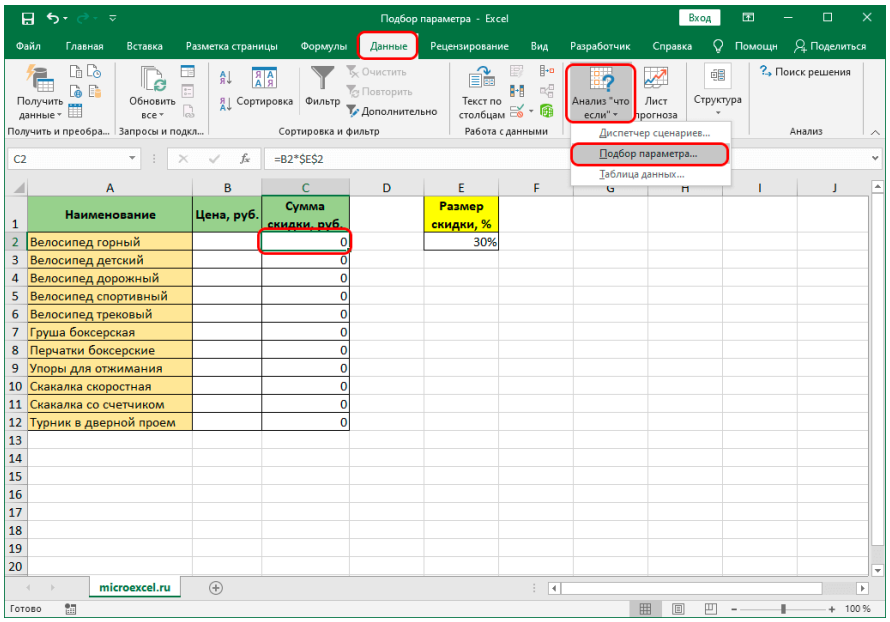
Давайте перейдем к практическому примеру, который позволит наилучшим образом понять, как работает функция.

Итак, у нас есть таблица с перечнем спортивных товаров. Мы знаем только сумму скидки (*560 руб. для первой позиции*) и ее размер, который для всех наименований одинаковый. Предстоит выяснить полную стоимость товара. При этом важно, чтобы в ячейке, в которой в дальнейшем отразится сумма скидки, была записана формула ее расчета (в нашем случае – умножение полной суммы на размер скидки).



Итак, алгоритм действий следующий:

1. Переходим во вкладку *“Данные”*, в которой нажимаем на кнопку *“Анализ “что если”* в группе инструментов *“Прогноз”*. В раскрывшемся списке выбираем *“Подбор параметра”* (в ранних версиях кнопка может находиться в группе *“Работа с данными”*).



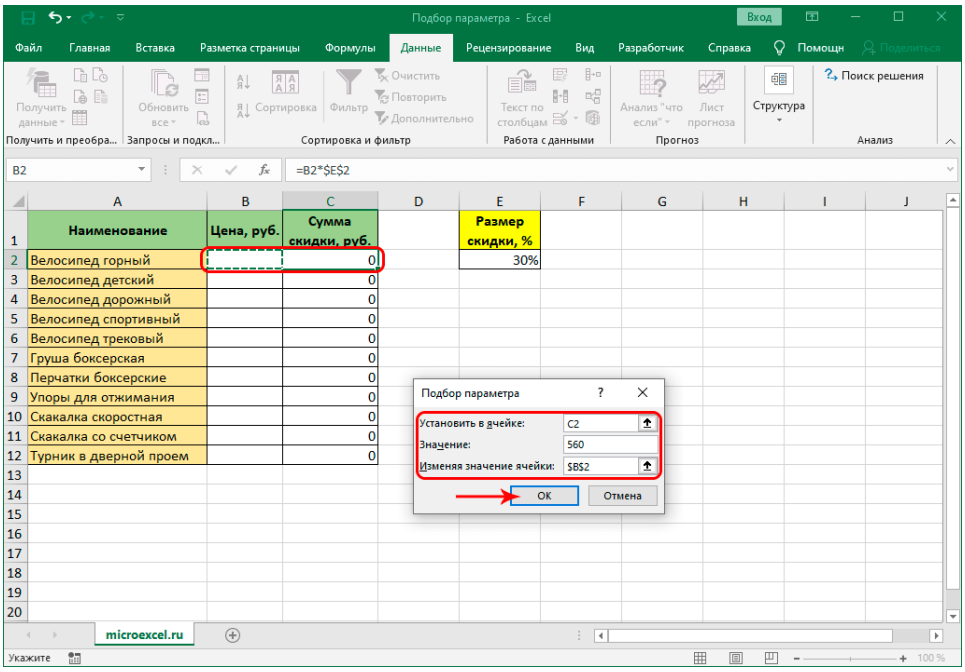
2. На экране появится окно для подбора параметра, которе нужно заполнить:

- В значении поля “Установить в ячейке” пишем адрес с финальными данными, которые нам известны, т.е. это ячейка с суммой скидки. Вместо ручного ввода координат можно просто щелкнуть по нужной ячейке в самой таблице. При этом курсор должен быть в соответствующем поле для ввода информации.

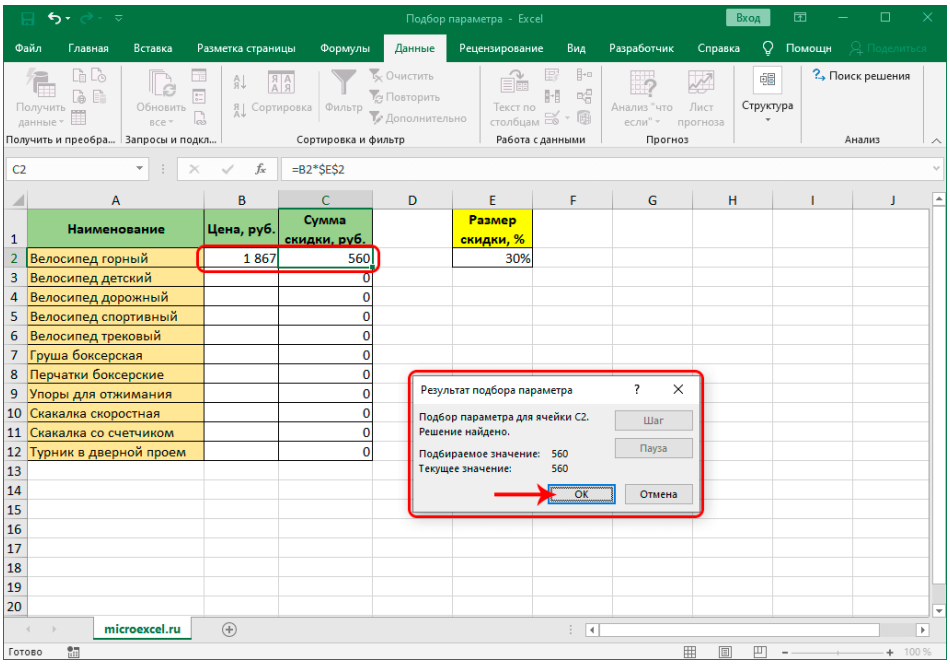
- В качестве значения указываем сумму скидки, которая нам известна – 560 руб.

- В поле *“Изменяя значение ячейки”* вручную или посредством клика мышью указываем координаты ячейки (должна участвовать в формуле расчета суммы скидки), в которой планируем вывести начальное значение.

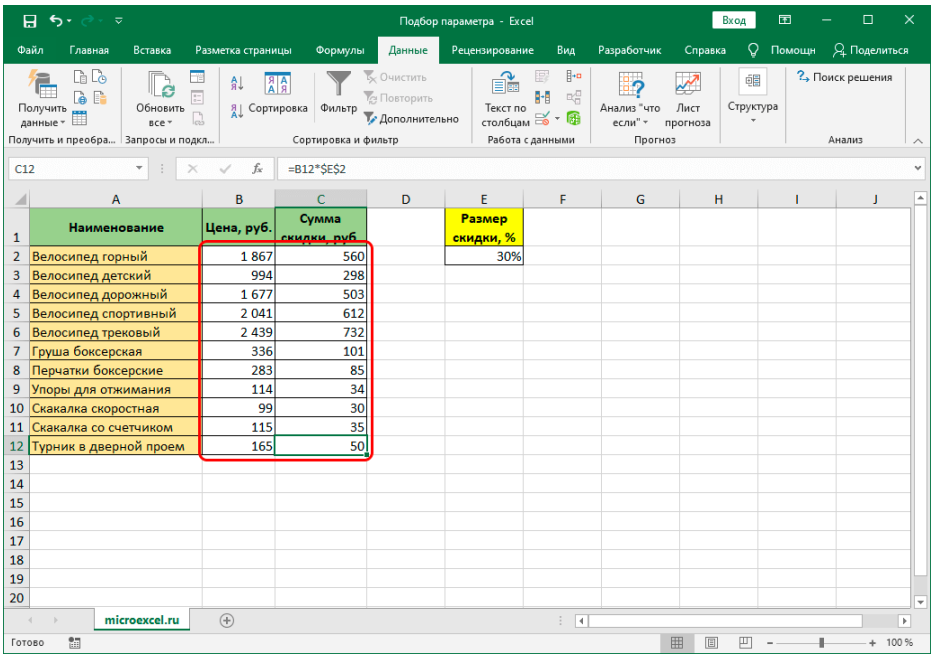
- по готовности нажимаем *ОК*.



3. Программа выполнит расчеты и выдаст результат в небольшом окошке, которое можно закрыть, нажав кнопку *OK*. Также найденные значения автоматически появятся в заданных ячейках таблицы.



4. Аналогичным образом можно посчитать цену без скидки для других товаров, если нам известна точная сумма скидки по каждому из них.

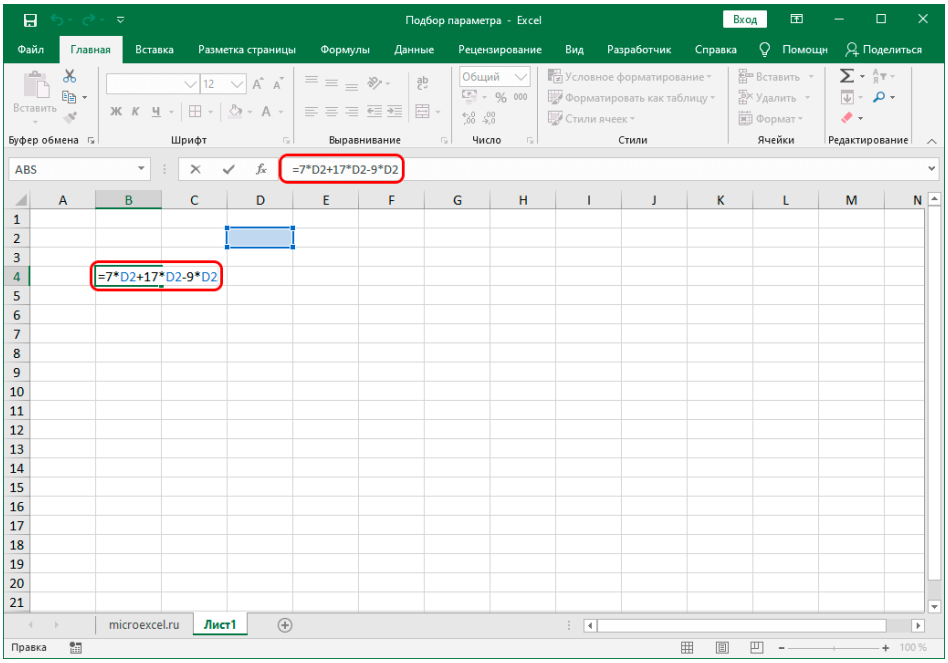


Решение уравнений с помощью подбора параметра

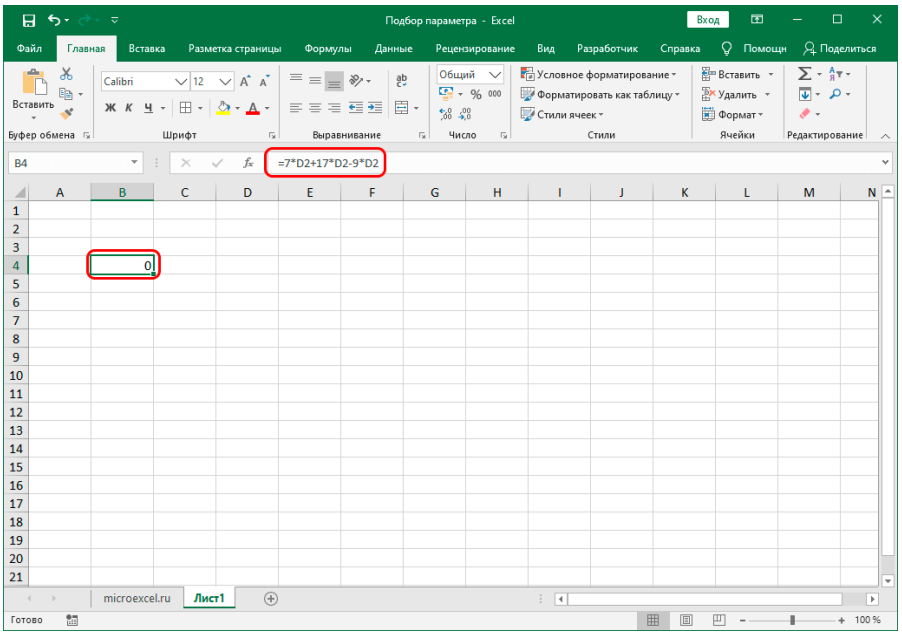
Несмотря на то, что это не основное направление использования функции, в некоторых случаях, когда речь идет про одну неизвестную, она может помочь в решении уравнений.

Например, нам нужно решить уравнение: 7x+17x-9x=75.

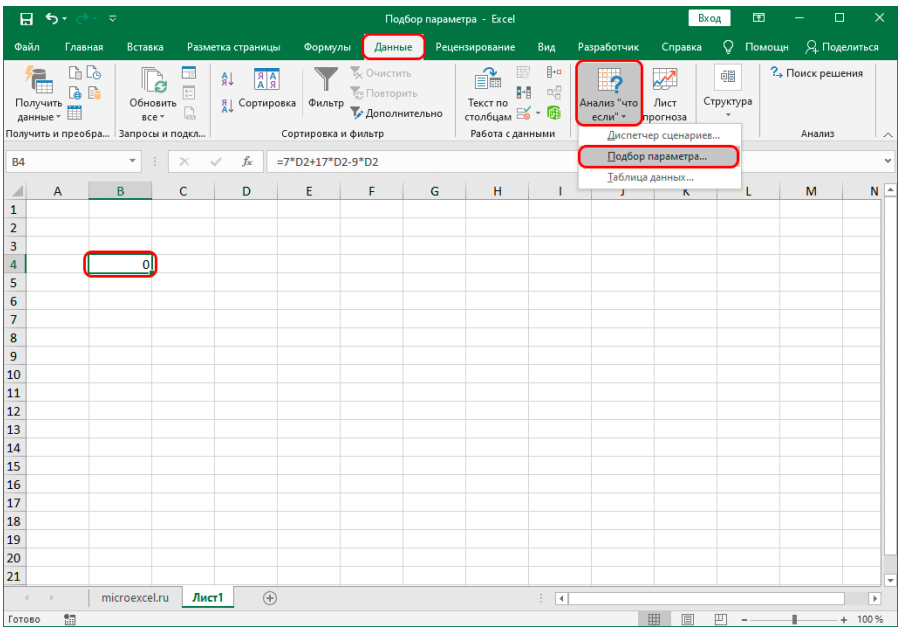
1. Пишем выражение в свободной ячейке, заменив символ ***x*** на адрес ячейки, значение которой нужно найти. В итоге формула выглядит так: =7*D2+17*D2-9*D2.



2. Щелкаем *Enter* и получаем результат в виде числа *0*, что вполне логично, так как нам только предстоит вычислить значение ячейки *D2*, которе и является “иксом” в нашем уравнении.

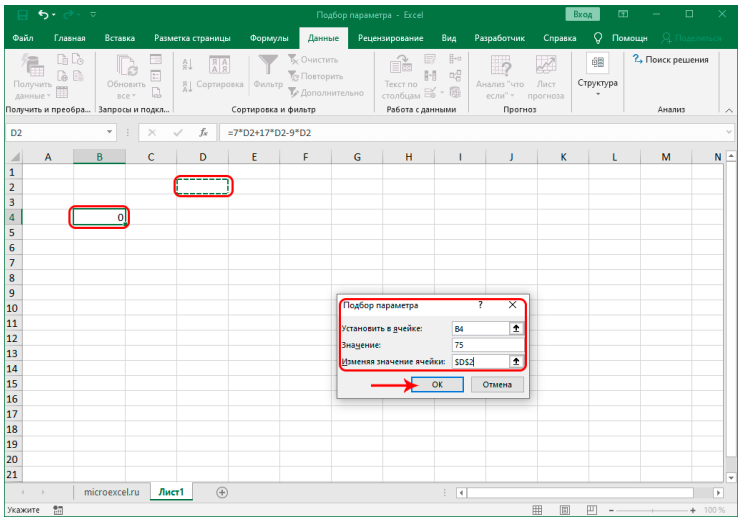


3. Как было описано в первом разделе статьи, во вкладке *“Данные”* нажимаем кнопку *“Анализ “что если”* и выбираем *“Подбор параметра”*.

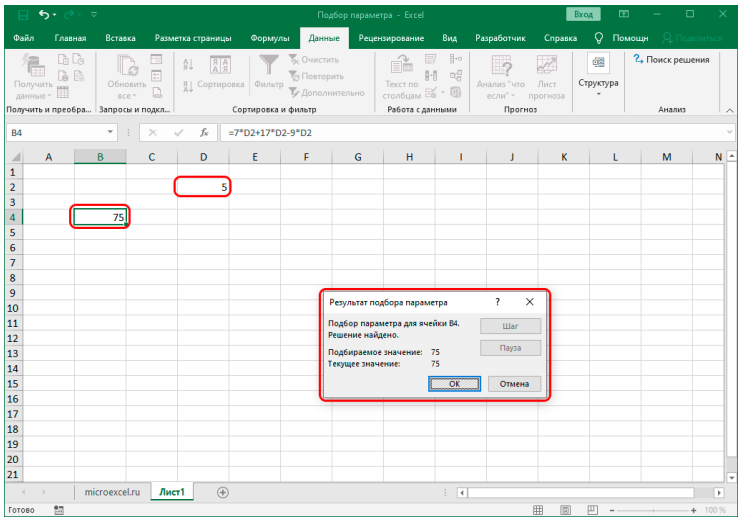


4. В появившемся окошке заполняем параметры:

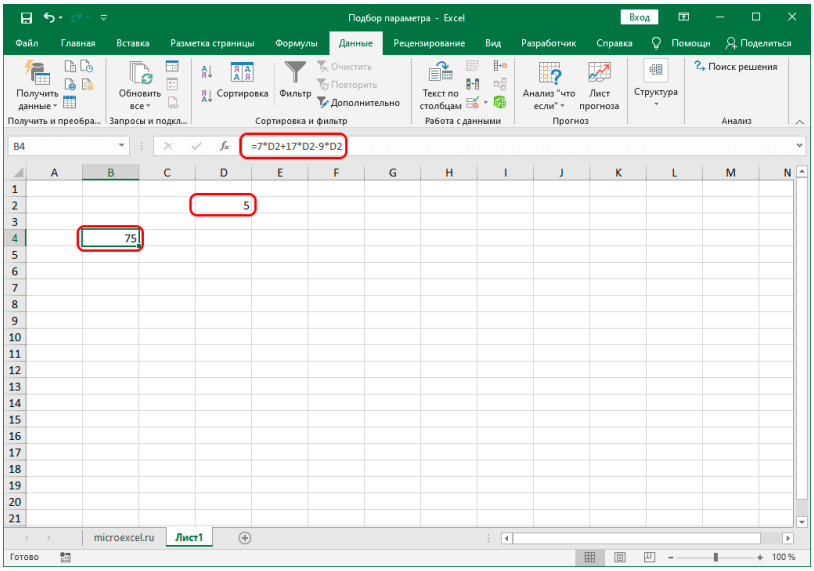
- В значении поля *“Установить в ячейке”* указываем координаты ячейки, в которой мы написали уравнение (т.е. *B4*).
- В значении, согласно уравнению, пишем число *75*.
- В поле *“Изменяя значения ячейки”* указываем координаты ячейки, значение которой нужно найти. В нашем случае – это *D2*.
- Когда все готово, нажимаем *OK*.



5. Как и в примере, рассмотренном выше, будут произведены вычисления и получен результат, о чем будет свидетельствовать небольшой окошко.



6. Таким образом, нам удалось решить уравнение и найти значение ***x***, которое оказалось равным числу 5.



Практическая часть

Задание 1. Рассчитать количество прожитых дней.

	A	B	C	D
1	Возраст учащихся			
2	Фамилия	Имя	Дата рождения	Возраст
3	Семенов	Саша	24.02.1986	
4	Замятина	Анна	21.09.1987	
5	Свиридова	Елена	23.02.1984	
6	Булавин	Михаил	13.08.1990	
7	Воронин	Женя	16.09.1986	
8	Егоров	Коля	14.08.1987	
9	Иванов	Олег	12.10.1988	
10	Новоселов	Петр	16.03.1986	
11	Арканов	Сергей	17.08.1986	
12	Петрова	Света	18.12.1986	
13	Иванова	Женя	19.08.1985	
14	Сидорова	Мария	20.08.1986	
15	Сорокина	Наталья	21.05.1986	
16	Суворов	Алексей	01.08.1987	
17	Рогожин	Иван	23.08.1986	
18	Удалов	Роман	24.11.1987	
19	Волошина	Светлана	25.08.1986	
20	Захарова	Ирина	26.01.1986	
21	Титов	Антон	27.08.1989	

Технология работы:

1. Запустить приложение Excel .
2. На **Листе 1** в ячейку А1 ввести дату своего рождения (число, месяц, год – 20.12.81). Зафиксируйте ввод данных.
3. Просмотреть различные форматы представления даты **(Главная-Формат – Формат ячейки – Число – Числовые форматы - Дата)**. Перевести дату в тип **ЧЧ.ММ.ГГГГ.** Пример,14.12.2016
4. Рассмотрите несколько типов форматов даты в ячейке А1.
5. В ячейку А2 ввести сегодняшнюю дату.
6. В ячейке А3 вычислить количество прожитых дней по формуле =А2-А1. Результат может оказаться представленным в виде даты, тогда его следует перевести в числовой тип. **(Главная - Формат – Формат ячейки – Число – Числовые форматы – Числовой – число знаков после запятой – 0**

Задание 2. Возраст студентов. По заданному списку студентов и даты их рождения. Определить, кто родился раньше (позже), определить кто самый старший (младший).

Технология работы:

1. Заполните на **Листе 2** таблицу по образцу.
2. Рассчитаем возраст студентов. Чтобы рассчитать возраст необходимо с помощью функции **СЕГОДНЯ** выделить сегодняшнюю текущую дату из нее вычитается дата рождения учащегося, далее из получившейся даты с помощью функции ГОД выделяется из даты лишь год. Из полученного числа вычтем 1900 – века и получим возраст учащегося. В ячейку D3 записать формулу **=ГОД(СЕГОДНЯ()-С3)-1900.** Результат может оказаться представленным в виде даты, тогда его следует перевести в **числовой тип. (Главная -Формат – Формат ячейки – Число – Числовые форматы – Числовой – число знаков после запятой – 0).**
3. Определим самый ранний день рождения. В ячейку С22 записать формулу =МИН(С3:С21);
4. Определим самого младшего учащегося. В ячейку D22 записать формулу =МИН(D3:D21);
5. Определим самый поздний день рождения. В ячейку С23 записать формулу =МАКС(С3:С21);
6. Определим самого старшего учащегося. В ячейку D23 записать формулу =МАКС(D3:D21).

Задача 3. Найти корень уравнения $x^3 - \cos x = 0$ приближенными методами.

Технология работы:

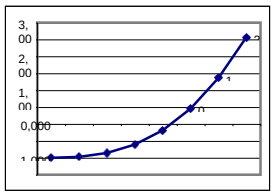
1. Представьте на **Листе 3** данное уравнение в табличной форме.

	A	B	C	D	E	F	G	H	I
1	x	0,000	0,200	0,400	0,600	0,800	1,000	1,200	1,400
2	$y = x^3 - \cos x$								
3									

2. Рассчитайте значение *y* по формуле: в ячейку B2 введите = B1^3-COS(B1)
3. Скопируйте данную формулу в остальные ячейки. В результате должна получиться таблица значений:

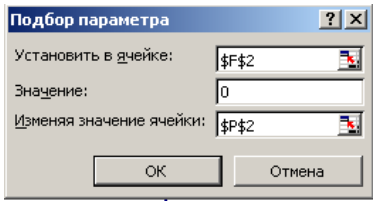
	A	B	C	D	E	F	G	H	I
1	x	0,000	0,200	0,400	0,600	0,800	1,000	1,200	1,400
2	$y = x^3 - \cos x$	-1,000	-0,972	-0,857	-0,609	-0,185	0,460	1,366	2,574
3									

4. Для приближенного определения корня построить диаграмму типа График. По графику приближенно можно определить, что $x \approx 0,8$



Для поиска решения с заданной точностью используем метод *Подбор параметра.* Точность подбора зависит от заданной точности представления чисел в ячейках таблицы (например, до трех знаков после запятой). Методом подбора параметра необходимо определить значение аргумента *x* (ячейка F1), при котором значение функции *y*(ячейка F2) будет равно нулю.

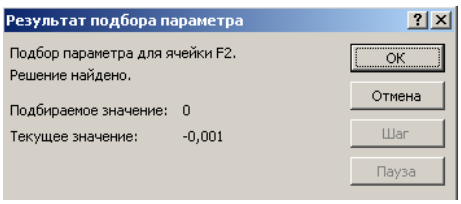
5. Выделить ячейку со значением функции F2 и ввести команду
Данные – Анализ «что если» - Подбор параметра



В диалоговом окне **Подбор параметра**: в поле **Установить в ячейке**: ввести адрес ячейки F2;в поле **Значение**: ввести требуемое значение функции, в данном случае 0;

6. в поле Изменяя значение ячейки: ввести адрес ячейки F1, в которой будет производиться подбор значения аргумента. Щелкнуть по кнопке ОК.

На панели Результат подбора параметра будет выведена информация о величине значения в ячейке F2.



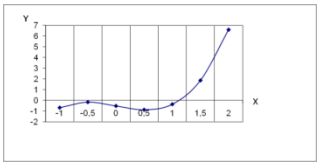
7. В ячейке аргумента F1 появится подобранное значение 0,865. Таким образом, корень уравнения $x \approx 0,865$ найден с заданной точностью.

Задача 4. Найти корень уравнения $x^3 - \sin x - 0,5 = 0$.

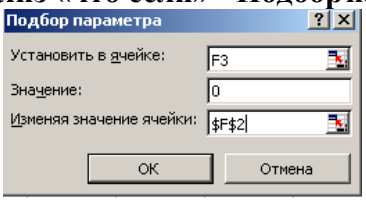
Технология работы:

1. Представьте функцию в табличной форме на **Листе 4**.

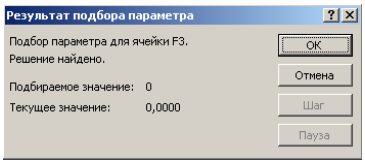
	A	B	C	D	E	F	G	H
1	*Таблица значений функции y=x^3-sinx-0,5*							
2	x	-1	-0,5	0	0,5	1	1,5	2
3	y	-0,6585	-0,1456	-0,5000	-0,8544	-0,3415	1,8775	6,5907
4								



2. Построить график функции. По графику приближенно можно определить, что уравнение имеет корень *x=1*.Методом подбора параметра необходимо определить значение аргумента x(ячейка F2), при котором значение функции y (ячейка F3) равна нулю. Ввести команду **Данные – Анализ «что если» - Подбор параметра**



На панели **Результат подбора параметра** будет выведена информация о величине подбираемого и подобранного значений. В ячейке F2 появится подобранное значение 1,1185. Таким образом, корень уравнения *x=1,1185* найден с заданной точностью.



Задача 5. На **Листе 5** самостоятельно выполните задание. Методом подбора параметрарешить уравнение x^2-*sin x + 0,1 =0* с точностью до четырех знаков после запятой.

Задание 6. Использование встроенных математических и статистических функций

1. Скопировать в свою папку файл ПР 19 из папки Файлы к практическому занятию №19
2. Открыть лист «Планеты» и выполнить следующие задания:
3. Скопировать лист «Планеты» на лист 2. Отсортировать по возрастанию столбец «Плотность».
4. Скопировать лист «Планеты» на лист 3. Отсортировать по убыванию столбец «Скорость вращения по орбите».
5. Скопировать лист «Планеты» на лист 4. Используя фильтры получить список планет, плотность которых более 3 и менее 5 г/см^3.
6. Скопировать лист «Планеты» на лист 5. Используя фильтры получить список планет, у которых температура поверхности отрицательная.
7. Скопировать лист «Планеты» на лист 6. Используя фильтры получить список планет, у которых скорость вращения по орбите больше 10 км/с, но меньше 25 км/с, а ускорение свободного падения не более 25 м/с^2.
8. Скопировать лист «Планеты» на лист 7. Используя фильтры получить список планет, у которых температура поверхности положительная и диаметр больше 9000 км.
9. Скопировать лист «Планеты» на лист 8. Используя фильтры получить список

планет, названия которых начинаются на буквы с А по Р. Отсортировать по алфавиту столбец «Планеты».

10. Перейти на лист **ведомость**.

№	Месяц	Отчетный год			Отклонение от плана
		план	фактически	выполнение, %	
i	Mi	Pi	Fi	Vi	Oi
1	январь	7 800,00р.	8 500,00р.	108,97%	700,00р.
2	февраль	3 560,00р.	2 700,00р.	75,84%	-860,00р.
3	март	8 900,00р.	7 800,00р.	87,64%	-1 100,00р.
4	апрель	5 460,00р.	4 590,00р.	84,07%	-870,00р.
5	май	6 570,00р.	7 650,00р.	116,44%	1 080,00р.
6	июнь	6 540,00р.	5 670,00р.	86,70%	-870,00р.
7	июль	4 900,00р.	5 430,00р.	110,82%	530,00р.
8	август	7 890,00р.	8 700,00р.	110,27%	810,00р.
9	сентябрь	6 540,00р.	6 500,00р.	99,39%	-40,00р.
10	октябрь	6 540,00р.	6 570,00р.	100,46%	30,00р.
11	ноябрь	6 540,00р.	6 520,00р.	99,69%	-20,00р.
12	декабрь	8 900,00р.	10 000,00р.	112,36%	1 100,00р.
	Максимум			116,44%	1 100,00р.
	Среднее			99,39%	40,83р.

11. Значения столбцов Oi и Vi вычисляются по формулам:Vi=Fi/Pi; Oi=Fi-Pi.

12. В эту таблицу снизу добавьте ячейки по образцу и выполните соответствующие вычисления (используйте функции МАКС(E4:E15) и СРЗНАЧ(E4:E15), МАКС(F4:F15) и СРЗНАЧ(F4:F15)).

13. Перейти на Лист **сведения о стаже сотрудников**. Заполните таблицу по образцу (чтобы расположить текст вертикально – п.к. мыши – выравнивание и надпись повернуть на 90 градусов ; в ячейках C3:C12 поставить формат ячеек – дата-длинный формат даты)

Сведения о стаже сотрудников фирмы "Рога и копыта"			
ФИО	Должность	Дата приема на работу	Стаж
Иванов И.И.	Директор	суббота, 01 Январь, 2005	13
Петров П.П.	Водитель	пятница, 02 Февраль, 2007	11
Сидоров С.С.	Инженер	суббота, 03 Июнь, 2006	11
Кошкин К.К.	Гл.Бух.	###	6
Мышкин М.М.	Охранник	четверг, 01 Август, 2013	4
Мошкин М.М.	Инженер	воскресенье, 04 Декабрь, 2005	12
Собакин С.С.	Техник	вторник, 06 Ноябрь, 2007	10
Лосев Л.Л.	Психолог	среда, 14 Апрель, 2010	7
Гусев Г.Г.	Техник	суббота, 25 Июль, 2009	8
Волков В.В.	Снабженец	вторник, 02 Май, 2006	11

14. Вычислить стаж работников по формуле: = ГОД(СЕГОДНЯ()-Дата приема на работу)-1900. (в ячейках D3:D12 поставьте формат ячеек – числовой-число знаков 0).

15. Перейдите не на лист **налоги**, измените заголовок таблицы, добавьте столбцы Ставка, Начислено, Налог, Заработная плата.

Заработная плата сотрудников фирмы "Рога и копыта"

ФИО	Должность	Дата приема на работу	Стаж	тарифные ставки	Ставка	Начислено	Налог	Заработная плата
Иванов И.И.	Директор	суббота, 01 Январь, 2005	13	2	5000	10000	2000	8000
Петров П.П.	Водитель	пятница, 02 Февраль, 2007	11	2	1000	2000	240	1760
Сидоров С.С.	Инженер	суббота, 03 Июнь, 2006	11	2	3000	6000	1200	4800
Кошкин К.К.	Гл.Бух.	###	6	1	4000	4000	800	3200
Мышкин М.М.	Охранник	четверг, 01 Август, 2013	4	1	3000	3000	600	2400
Мошкин М.М.	Инженер	воскресенье, 04 Декабрь, 2005	12	2	4000	8000	1600	6400
Собакин С.С.	Техник	вторник, 06 Ноябрь, 2007	10	2	2000	4000	800	3200
Лосев Л.Л.	Психолог	среда, 14 Апрель, 2010	7	1	3000	3000	600	2400
Гусев Г.Г.	Техник	суббота, 25 Июль, 2009	8	2	500	1000	120	880
Волков В.В.	Снабженец	вторник, 02 Май, 2006	11	2	3500	7000	1400	5600

Столбец Тарифные ставки и вычислить таким образом: 1 – если стаж меньше 8 лет, 2 – если стаж больше 8 лет (=ЕСЛИ(D3<8;1;2))

Начислено=Ставка*Тарифные ставки.

Налог если Начислено меньше 3000, 12%, если Начислено больше 3000 20% (=ЕСЛИ(G3<3000;12%;20%)*G3)

Заработная плата Начислено - Налог

16. Сохранить в папке под вашим именем, файл назовите Встроенные функции.
17. Текстовые функции

1. Расположите в ячейках A1, B1, C1 соответственно тексты: МУ, КОШКА, РОВ.
2. Скопируйте эти значения вниз на 5 строк.
3. Поместите курсор в ячейку D1. В текстовых функциях выберите ПОВТОР, установите – 3 раза для ячейки A1.
4. Теперь получим из двух слов новое. Поместите курсор в ячейку D2. В текстовых функциях выберите ЗАМЕНИТЬ. Старый текст – B2, начальная позиция – 3, число знаков – 1 (сколько знаков будут заменены), новый текст – C2 (может быть слово).
5. Поместите курсор в ячейку D3. Выберите функцию СЦЕПИТЬ. Текст 1 – A3, текст 2 – B3.
6. Заменим слово «ров» на «рога». Поместите курсор в ячейку D4. Выберите функцию ЗАМЕНИТЬ. Текст – C4, позиция — 3, длина -1, новый текст – «ГА».
7. Теперь выберем нужное количество букв из слова. Поместите курсор в ячейку D5. Выберите функцию ПСТР. Текст – D2, начальная позиция – 3, количество знаков – 5.

Заработная плата сотрудников фирмы "Рога и копыта"								
ФИО	Должность	Дата приема на работу	Стаж	тарифные ставки	Ставка	Начислено	Налог	Заработная плата
Иванов И.И.	Директор	суббота, 01 Январь, 2005	13	2	5000	10000	2000	8000
Петров П.П.	Водитель	пятница, 02 Февраль, 2007	11	2	1000	2000	240	1760
Сидоров С.С.	Инженер	суббота, 03 Июнь, 2006	11	2	3000	6000	1200	4800
Кошкин К.К.	Гл.Бух.	###	6	1	4000	4000	800	3200
Мышкин М.М.	Охранник	четверг, 01 Август, 2013	4	1	3000	3000	600	2400
Мошкин М.М.	Инженер	воскресенье, 04 Декабрь, 2005	12	2	4000	8000	1600	6400
Собакин С.С.	Техник	вторник, 06 Ноябрь, 2007	10	2	2000	4000	800	3200
Лосев Л.Л.	Психолог	среда, 14 Апрель, 2010	7	1	3000	3000	600	2400
Гусев Г.Г.	Техник	суббота, 25 Июль, 2009	8	2	500	1000	120	880
Волков В.В.	Снабженец	вторник, 02 Май, 2006	11	2	3500	7000	1400	5600

Начислено=Ставка*Тарифные ставки.
Налог если Начислено меньше 3000, 12%, если Начислено больше 3000 20%
(=ЕСЛИ(G3<3000;12%;20%)*G3)
Заработная плата Начислено - Налог
4.Сохранить в папке под именем Встроенные функции

Контрольные вопросы:

1. Как выглядит указатель мыши в процессе выделения блока (диапазона) ячеек?
2. Как выглядит указатель мыши в процессе автозаполнения ячеек?
3. Что такое смарт-тег? Найдите ответ в справочной системе к программе MS Excel.
4. Как изменить ширину столбца двумя способами?
5. Как вычислить сумму нескольких значений в столбце?
6. Как придать числу денежный стиль?
7. Как обрамить ячейки?
8. Как придать ячейкам фон?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 20

Тема: Наглядное представление результатов статистической обработки данных в виде диаграмм средствами редактора электронных таблиц

Количество часов на выполнение:6

Цель работы: изучить технологию построения диаграмм в ЭТ, научиться правильно указывать все необходимые параметры при построении графиков функций и диаграмм, использовать графические объекты для анализа данных.

Оборудование: Персональный компьютер

Методические указания к выполнению:
Теоретическая часть

Диаграмма – удобное средство графического представления данных. Позволяет оценить имеющиеся величины. Для наглядного представления данных необходимо определить **Тип** и **Вид** диаграммы.

Гистограмма – иллюстрирует соотношение отдельных значений данных. Категории располагаются по горизонтали, а значения по вертикали. 1 столбец – позволяет сравнивать показатели величин друг с другом. 2 столбец – показывает численную долю отдельного показателя в суммарном объеме. 3 столбец – позволяет сравнивать процентное соотношение долей каждого показателя в общем объеме.

Круговая диаграмма – показывает вклад элемента в общую сумму. Всегда строится по одному ряду данных.

График – показывает зависимость одной величины от другой.

Лепестковая диаграмма – позволяет сравнивать общие значения из нескольких наборов данных. Каждая категория имеет собственную ось координат, исходящую из начала координат.

Конусная, пирамидальная, поверхностная диаграммы - позволяют разнообразить типы диаграмм.

Практическая часть

Задание 1. Заполнить и настроить таблицу для расчета времени, затраченного разными животными при движении на заданное расстояние. Построить гистограмму по образцу.

Технология выполнения задания

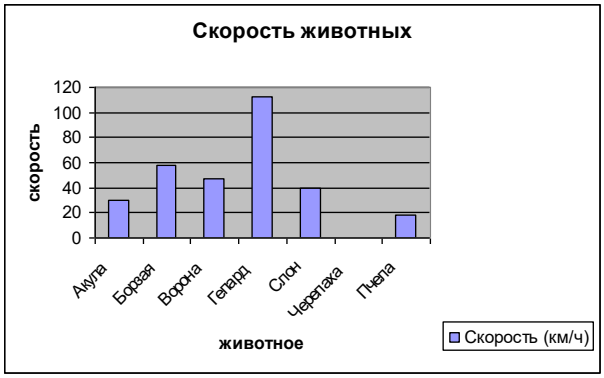
Исходные данные – задаются с клавиатуры:

- Столбец B - значение скорости животного (в км/ч).
- Столбец C – расстояние, которое надо преодолеть (в км).

Вычисляемые значения – считаются по формуле:

- Столбец D – время.
- Ячейка C11 – максимальная скорость (функция МАКС).
- Ячейка C12 – минимальная скорость (функция МИН).

	A	B	C	D
1	**Животное**	**Скорость (км/ч)**	**Расстояние (км)**	**Время (ч)**
2	Акула	30	100	3
3	Борзая	58	100	2
4	Ворона	47	100	2
5	Гепард	112	100	1
6	Слон	40	100	3
7	Черепаха	0,5	100	200
8	Пчела	18	100	6
9				
10	**Статистические данные**			
11	Максимальная скорость	112		
12	Минимальная скорость	0,5		
13				



Задание 2. Заполнить и настроить таблицу для расчета основных характеристик *Мирового океана*. Построить круговую диаграмму по образцу.

Технология выполнения задания

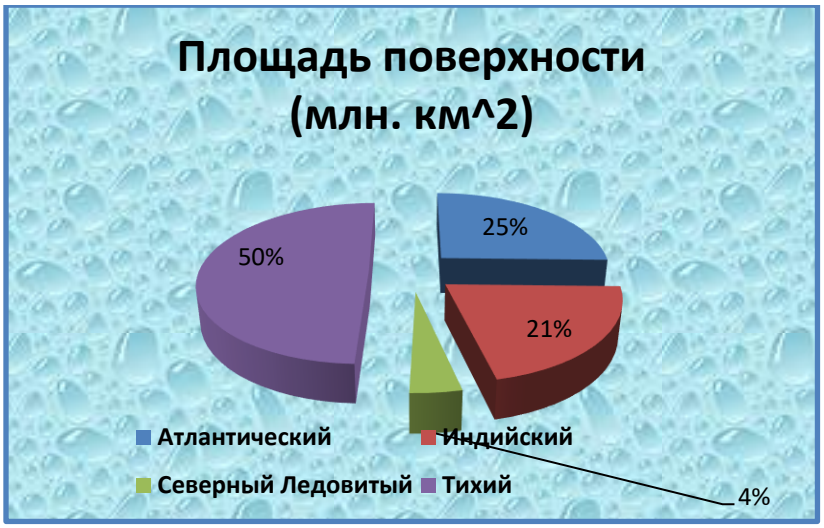
Исходные данные – задаются с клавиатуры:

- Диапазон A1:E5.

Вычисляемые значения – считаются по формуле:

- Ячейка B6 – общая площадь Мирового океана (функция СУММ).
- Ячейка C6 – общий объем Мирового океана (функция СУММ).
- Ячейка D6 – средняя глубина Мирового океана (функция СРЗНАЧ).
- Ячейка E6 – самая глубокая впадина Мирового океана (функция МАКС).

	A	B	C	D	E
1	**Океаны**	**Площадь поверхности (млн. км^2)**	**Объем (млн. км^3)**	**Средняя глубина (м)**	**Наибольшая глубина (м)**
2	Атлантический	91,66	329,66	3597	8742
3	Индийский	76,17	282,65	3711	7209
4	Северный Ледовитый	14,75	18,07	1225	5527
5	Тихий	178,68	710,36	3976	11022
6	Мировой	361,26	1340,74	3127,25	11022
7					



Задание 3. Заполнить и настроить таблицу *Рейтинг самых высоких гор по континентам*. Построить график по образцу.

Технология выполнения задания

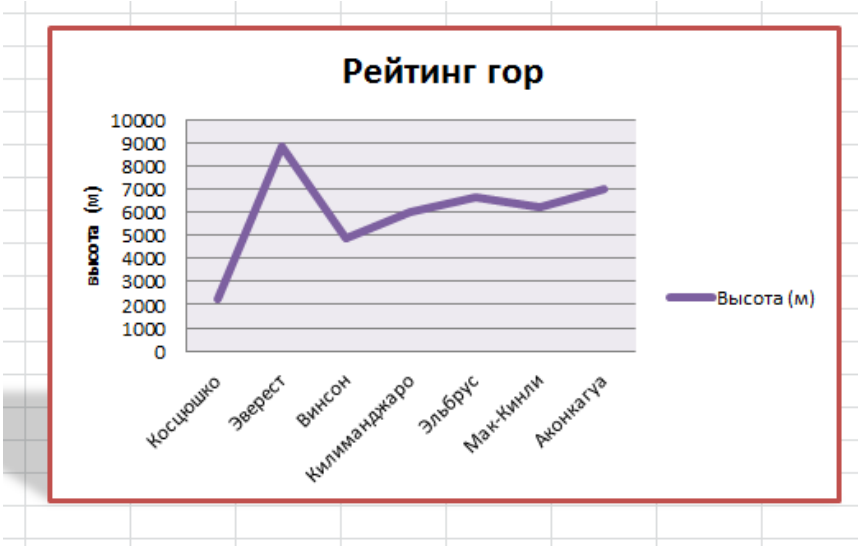
Исходные данные – задаются с клавиатуры:

- Диапазон A1:C8.

Вычисляемые значения – считаются по формуле:

- Ячейка B11 – средняя высота заданных вершин (функция СРЗНАЧ).
- Ячейка B12 – самая высокая гора в мире (функция МАКС).
- Ячейка B13 – самая низкая из самых высоких гор (функция МИН).

	A	B	C
1	**Континет**	**Самая высокая гора**	**Высота (м)**
2	Австралия	Косцюшко	2226
3	Азия	Эверест	8852
4	Антарктида	Винсон	4892
5	Африка	Килиманджаро	5995
6	Европа	Эльбрус	6642
7	Северная Америка	Мак-Кинли	6194
8	Южная Америка	Аконкагуа	6962
9			
10	**Статистические данные**		
11	Средняя высота гор	**5966**	
12	Самая высокая гора	**8852**	
13	Самая низкая гора	**2226**	
14			



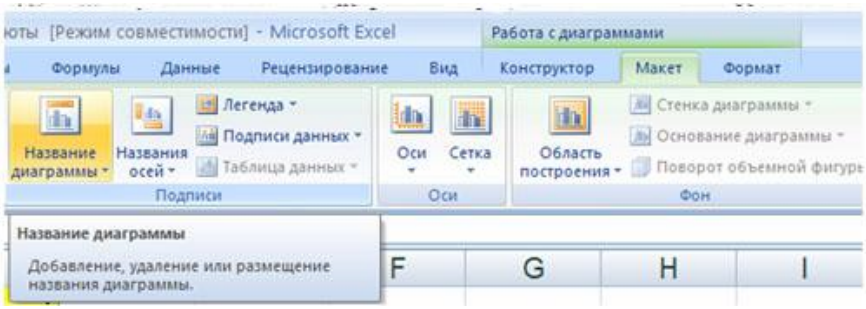
Задание 4. Построить графики функций ***y=sin(x)*** и ***y=sin(2x)*** на интервале ***[- 180; 160]*** с шагом ***20.***

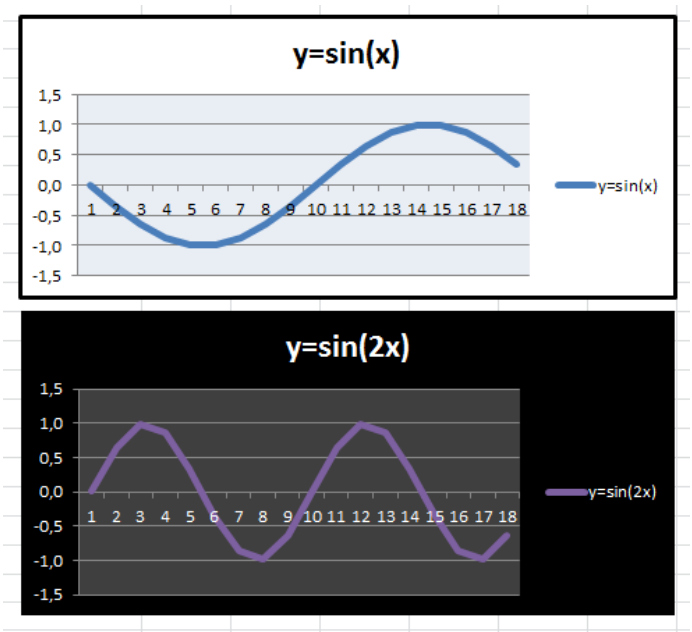
Технология выполнения задания

1. Заполнить таблицу значений:

X (град.)	X (рад.)	y=sin(x)	y=sin(2x)
-180	-3,1	0,0	0,0
-160	-2,8	-0,3	0,6
-140	-2,4	-0,6	1,0
-120	-2,1	-0,9	0,9
-100	-1,7	-1,0	0,3
-80	-1,4	-1,0	-0,3
-60	-1,0	-0,9	-0,9
-40	-0,7	-0,6	-1,0
-20	-0,3	-0,3	-0,6
0	0,0	0,0	0,0
20	0,3	0,3	0,6
40	0,7	0,6	1,0
60	1,0	0,9	0,9
80	1,4	1,0	0,3
100	1,7	1,0	-0,3
120	2,1	0,9	-0,9
140	2,4	0,6	-1,0
160	2,8	0,3	-0,6

2. Выделить таблицу и указать тип диаграммы График.
3. Выбрать формат графика с гладкими кривыми.
4. В Макете указать название диаграммы по образцу, разместить легенду справа, применить формат графика по образцу.





Задание 5. Произвести расчеты и построить гистограмму, круговую и линейчатую диаграммы.

Технология выполнения задания

Исходные данные – задаются с клавиатуры:

- Диапазон A3:A10, B3:B10.

Вычисляемые значения – считаются по формуле:

- Ячейка C5 – посчитать премию в размере 25% от оклада по формуле =B5*C4 и произвести автозаполнение до ячейки C11.
- Ячейка D5 – посчитать районный коэф. в размере 15% от оклада по формуле = B5*D4 и произвести автозаполнение до ячейки D11.
- Ячейка E5 - посчитать НДФЛ в размере 13% от оклада по формуле =(B5+C5+D5)*E4 и произвести автозаполнение до ячейки E11.
- Сравнить результаты с исходной таблицей.
- Построить диаграммы для диапазона данных столбцов ФИО и К выплате по образцу.

1	**Произвести расчеты и построить гистрограмму, круговую и линейчатую диаграммы**					
2						
3	**ФИО**	**Оклад**	**Премия**	**Районный коэф.**	**НДФЛ**	**К выплате**
4			**25%**	**15%**	**13%**	
5	**Иванов И.А.**	**10000**	**2500**	**1500**	**1820**	**12180,0**
6	**Петров Д.С.**	**15000**	**3750**	**2250**	**2730**	**18270,0**
7	**Васильева Е.С.**	**85000**	**21250**	**12750**	**15470**	**103530,0**
8	**Галкин Л.П.**	**9800**	**2450**	**1470**	**1783,6**	**11936,4**
9	**Романова И.С.**	**23000**	**5750**	**3450**	**4186**	**28014,0**
10	**Николаев С.П.**	**12000**	**3000**	**1800**	**2184**	**14616,0**
11						

Контрольные вопросы

1. Что называется деловой графикой?
2. Перечислить основные типы диаграмм?
3. В чем разница между круговой диаграммой и гистограммой?
4. Как построить диаграммы по числовым данным?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 21

Тема: Решение задач оптимизации с помощью электронных таблиц.

Количество часов на выполнение: 4

Цель работы: Изучение специальных возможностей электронных таблиц для решения оптимизационных задач и научиться создавать компьютерную модель для решения таких задач.

Оборудование: Персональный компьютер, MS Excel

Методика выполнения задания
Теоретическая часть

Ежедневно специалисты в области экономики и менеджмента сталкиваются с задачами оптимизации. Это и премирование штатного расписания, и расчет фонда заработной платы, и планирование рекламной кампании, и еще множество задач,

решаемых с помощью методов оптимизации. Наиболее легкими и показательными являются задачи линейной оптимизации.

Линейное программирование – это раздел высшей математики, занимающийся разработкой методов поиска экстремальных значений линейной функции, на неизвестные которой наложены линейные ограничения.

Задачи линейного программирования относятся к задачам на условный экстремум функции. Однако для исследования линейной функции многих переменных на условный экстремум нельзя применить хорошо разработанные методы математического анализа.

Отсюда $c_j = 0, (j = \overline{1,n})$. Так как все коэффициенты линейной функции не могут быть равны нулю, то внутри области, образованной системой ограничений, экстремальные точки не существуют. Они могут быть только на границе области.

Для решения таких задач разработаны специальные методы линейного программирования, которые особенно широко применяются в экономике.

Пример линейной оптимизационной задачи.

Для производства столов и шкафов мебельная фабрика использует необходимые ресурсы. Нормы затрат ресурсов на одно изделие данного вида, прибыль от реализации одного изделия и общее количество имеющихся ресурсов каждого вида приведены в табл. 1.

Таблица 1

Ресурсы	Нормы затрат ресурсов на одно изделие		Общее количество ресурсов
	Стол	Шкаф	
Древесина вид 1	0,2	0,1	40
Древесина вид 2	0,1	0,3	60
Трудоемкость (человеко-часов)	1,2	1,5	371,4
Прибыль от реализации одного изделия (руб.)	6	8	

Определить, сколько столов и шкафов фабрике следует изготовлять, чтобы прибыль от их реализации была максимальной.

Решение

1.Для решения этой задачи необходимо построить математическую модель.

2.Процесс построения модели можно начать с ответа на следующие три вопроса:

3.Для определения каких величин строится модель?

В чем состоит цель, для достижения которой из множества всехдопустимых значений переменных выбираются оптимальные?

Каким ограничениям должны удовлетворять неизвестные?

В данном случае мебельной фабрике необходимо спланировать объем производства столов и шкафов так, чтобы максимизировать прибыль. Поэтому переменными являются: x_1 – количество столов, x_2 – количество шкафов.

Суммарная прибыль от производства столов и шкафов равна $z = 6 \cdot x_1 + 8 \cdot x_2$. Целью фабрики является определение среди всех допустимых значений x_1 и x_2 таких, которые максимизируют суммарную прибыль, т.е. целевую функцию z.

Ограничения, которые налагаются на x_1 и x_2:

объем производства шкафов и столов не могут быть отрицательным, следовательно x_1, x_2.

нормы затрат древесины на столы и шкафы не могут превосходить максимально возможный запас данного исходного продукта, следовательно

$$0.2x_1 + 0.1x_2 \leq 40$$

$$0.1x_1 + 0.2x_2 \leq 60$$

Кроме того, ограничение на трудоемкость не превышает количества затрачиваемых ресурсов:

$$1.2x_1 + 1.5x_2 \leq 371.4$$

Таким образом, математическая модель данной задачи имеет следующий вид:

максимизировать функцию

$$z = 6 \cdot x_1 + 8 \cdot x_2$$

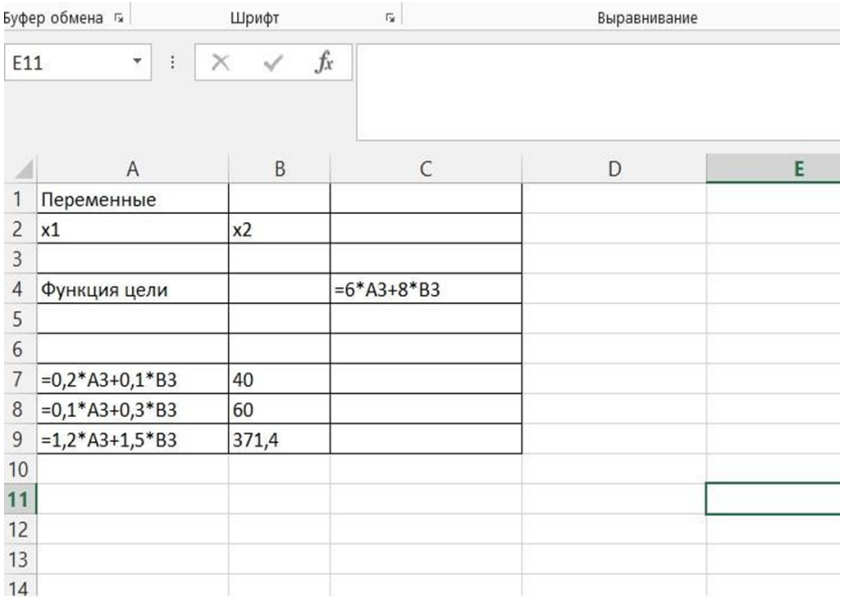
при следующих ограничениях:

$$\begin{cases} 0.2x1 + 0.1x2 \leq 40 \\ 0.1x1 + 0.2x2 \leq 60 \\ 1.2x1 + 1.5x2 \leq 371.4 \end{cases}$$

Данная модель является линейной, т.к. целевая функция и ограничения линейно зависят от переменных.

Решение задачи с помощью MS Excel

1. Отвести ячейки А3 и В3 под значения переменных x1 и x2 (Рисунок - 1)



	A	B	C	D	E
1	Переменные				
2	x1	x2			
3					
4	Функция цели		=6*A3+8*B3		
5					
6					
7	=0,2*A3+0,1*B3	40			
8	=0,1*A3+0,3*B3	60			
9	=1,2*A3+1,5*B3	371,4			
10					
11					
12					
13					
14					

Рисунок - 1. Диапазоны, отведенные под переменные, целевую функцию и ограничения.

2. В ячейку С4 ввести функцию цели: =6*A3+8*B3, в ячейки А7:А9 ввести левые части ограничений:

=0,2*A3+0,1*B3
=0,1*A3+0,3*B3
=1,2*A3+1,5*B3, а в ячейки B7:B9 – правые части ограничений (Рисунок - 1).

3. Выбрать вкладку «Данные». В разделе анализ выбрать команду «Поиск решения» и заполнить открывшееся диалоговое окно «Параметры поиска решения» так, как показано на Рисунок - 2. Средство поиска решений является одной из надстроек Excel. Если в меню «Данные» отсутствует команда «Поиск решения», то для её установки необходимо выполнить команду Файл/ Параметры/ Надстройки/ и подключить панель «Поиск решения». Для ввода ограничений

4. нажмите кнопку Добавить (рисунок 2)

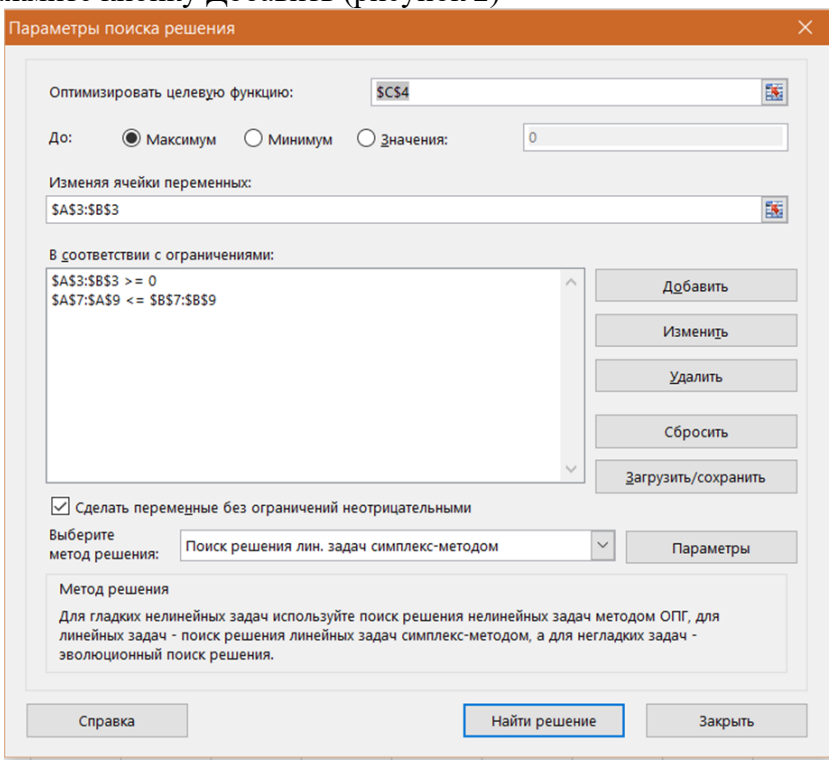


Рисунок - 2.Диалоговое окно: Параметры поиска решения задачи о максимизации прибыли на фабрике.

Внимание! Необходимо выбрать метод решения. Для решения линейной задачи выберите «поиск решения лин. задач симплекс-методом» (см. Рисунок - 2).

Выберите команду «Параметры», в открывшимся окне «Параметры» заполните максимальное время в секундах и число итераций как на Рисунок - 3.

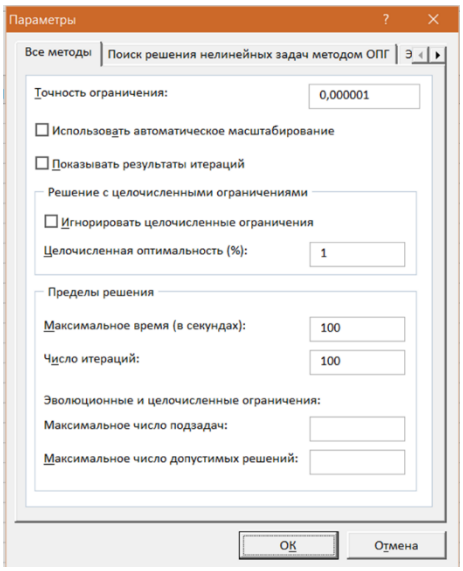


Рисунок - 3. Диалоговое окно Параметры поиска решения

5. После нажатия кнопки Найти решение открывается окно Результаты поиска решения, которое сообщает, что решение найдено (Рисунок - 4).

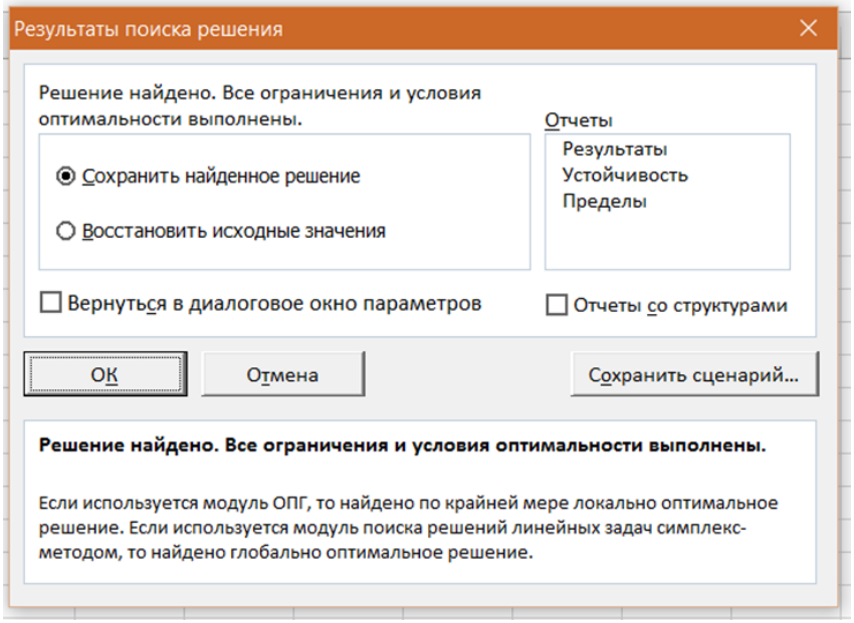


Рисунок - 4. Диалоговое окно Результаты поиска решения

6. Результаты расчета задачи представлены на Рисунок - 5, из которого видно, что оптимальным является производство 102 столов и 166 шкафов. Этот объем производства принесет фабрике 1940 руб. прибыли.

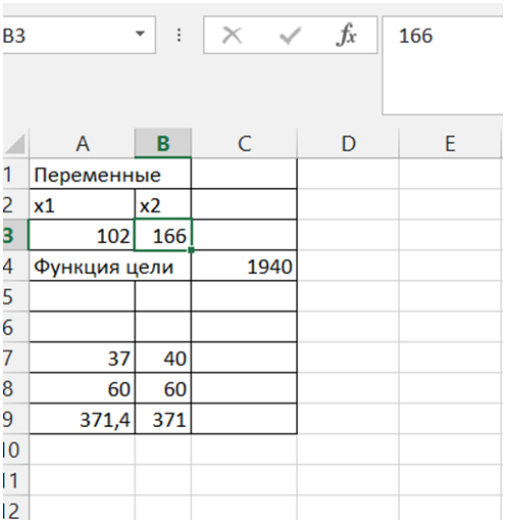


Рисунок - 2.5. Результаты расчета с помощью средства поиска решений для задачи максимизации выпуска столов и шкафов.

Практическая часть

Задание 1. Построить математическую модель задачи. Решить задачу с помощью средства MS Excel Поиск решения. Сделать соответствующие выводы.

Для производства двух видов изделий А и В используется токарное, фрезерное и шлифовальное оборудование. Нормы затрат времени для каждого из типов оборудования на одно изделие данного вида приведены в табл. 2.2. В ней же указан общий фонд рабочего времени каждого из типов оборудования, а также прибыль от реализации одного изделия.

Таблица 2

Тип оборудования	Затраты времени (станко-часов) на обработку одного изделия		Общий фонд полезного рабочего времени
	A	B	
Фрезерное Токарное Шлифовальное	10 5 6	8 10 12	168 180 144
Прибыль от реализации одного изделия (руб.)	14	18	

Определить план выпуска изделий вида А и В, обеспечивающий максимальную прибыль от их реализации.

Задание 2. Построить математическую модель задачи. Решить задачу с помощью средства MS Excel Поиск решения. Сделать соответствующие выводы.

Для изготовления различных изделий А, В и С предприятие использует три разных вида сырья. Нормы расхода сырья на производство одного изделия каждого вида, цена одного изделия А, В и С, а также общее количество сырья каждого вида, которое может быть использовано предприятием, приведены в табл. 3.

Таблица 3

Вид сырья	Норма затрат сырья (кг) на одно изделие	Общее количество

	A	B	C	сырья (кг)
Вид 1 Вид 2 Вид 3	18 6 5	15 4 3	12 8 3	360 192 180
Цена одного изделия (руб.)	9	10	16	

Изделия A, B и C могут производиться в любых соотношениях (сбыт обеспечен), но производство ограничено выделенным предприятию сырьем каждого вида.

Составить план производства изделий, при котором общая стоимость всей произведенной предприятием продукции является максимальной.

Задание 3. Построить математическую модель задачи. Решить задачу с помощью средства MS Excel Поиск решения. Сделать соответствующие выводы.

На швейной фабрике для изготовления четырех видов изделий может быть использована ткань трех артикулов. Нормы расхода тканей всех артикулов на пошив одного изделия приведены в табл. 4. В ней же указаны имеющиеся в распоряжении фабрики общее количество тканей каждого артикула и цена одного изделия данного вида.

Таблица 4

Артикул ткани	Норма расхода ткани (м) на одно изделие вида				Общее количество ткани (м)
	Вид 1	Вид 2	Вид 3	Вид 4	
Артикул 1 Артикул 2 Артикул 3	1 - 4	- 1 2	2 3 -	1 2 4	180 210 800
Цена одного изделия (руб.)	9	6	4	7	

Определить, сколько изделий каждого вида должна произвести фабрика, чтобы стоимость изготовленной продукции была максимальной.

Вариант 4

Фабрика «GRM pie» выпускает два вида каш для завтрака – «Crunchy» и «Chewy». Используемые для производства обоих продуктов ингредиенты в основном одинаковы и, как правило, не являются дефицитными.

Основным ограничением, накладываемым на объем выпуска, является наличие фонда рабочего времени в каждом из трех цехов фабрики.

Управляющему производством Джою Дисону необходимо разработать план производства на месяц. В табл. 5 указаны общий фонд рабочего времени и число человеко-часов, требуемое для производства 1 т продукта.

Контрольные вопросы

1. Какого типа задачи могут быть решены с помощью линейного программирования?
2. Что понимается под оптимальным решением?
3. Что такое условный экстремум функции?
4. Что такое целевая функция?
5. При каких условиях математическую модель можно назвать линейной?
6. Опишите процесс решения задачи линейного программирования

средствами MS Excel.

7. Опишите процесс решения средствами транспортной задачи при использовании Поиск решения MS Excel.
8. В чем отличие функций минимизации и максимизации при их задании в Поиске решения MS Excel?
9. Перечислите отличительные особенности решения транспортной задачи.
10. Опишите процесс формирования системы ограничений при решении задач линейного программирования

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 22

Тема: Разработка многотабличной базы данных.

Количество часов на выполнение: 4.

Цель работы: научится создавать многотабличные базы данных в программе Microsoft Access 2010

Оборудование: Персональный компьютер, ОС Windows, Microsoft Word, Microsoft Access 2010

Методика выполнения задания

Теоретическая часть

Постановка задачи. Создать базу данных «Библиотека» содержащую информацию о книгах, взятых читателями в библиотеке.

Выполнение работы.

Создадим новую базу данных в MS Access 2010, для этого зададим команду Файл/Создать и в окне Имя файла введем имя нашей БД

«Библиотека», далее нажимаем на командную кнопку Создать.

Создадим таблицу «Читатель» по команде Создание/Таблица, содержащую следующую информацию о читателях: Фамилия, имя, Домашний адрес, Номер паспорта, Телефон, (см. Рисунок - 2). В эту таблицу нам нужно добавить новое поле Код читателя для создания ключевого поля, потому что ни одно из представленных выше не подходить для этого. Для создания таблицы «Читатель» в MS Access 2010 зададим следующие команды:

Для ввода режима таблицы Конструктор: Поля/ раздел Режимы/ Конструктор. В появившемся окне

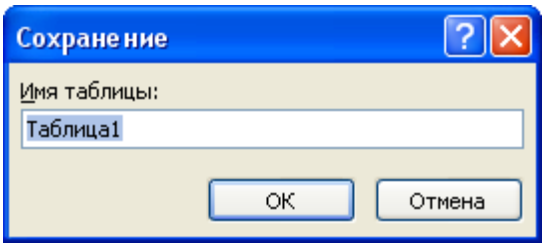


Рисунок - 1 Окно Сохранение

В поле Имя таблицы ввести имя таблицы Читатель.

В режиме конструктор, вводим имена полей в столбец Имя поля выбрав соответствующие для каждого тип данных как показано на Рисунок - 2. На рисунке видно, что в качестве ключевого поля мы выбрали поле Код читателя.

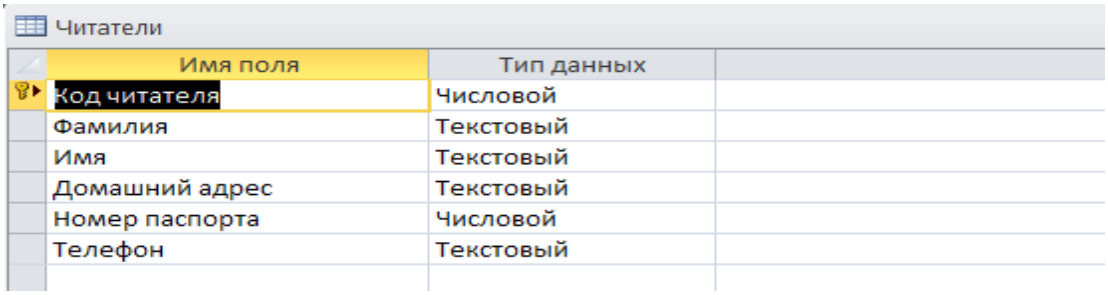


Рисунок - 2 Ввод таблицы в режиме конструктора

Для поля Телефон создадим маску ввода. Например, покажем, как создать маску ввода для телефона. Ставим курсор на поле Телефон как показано на рис 3. и в окне Свойства поля выбираем команду маска ввода, далее нажимаем троеточие 

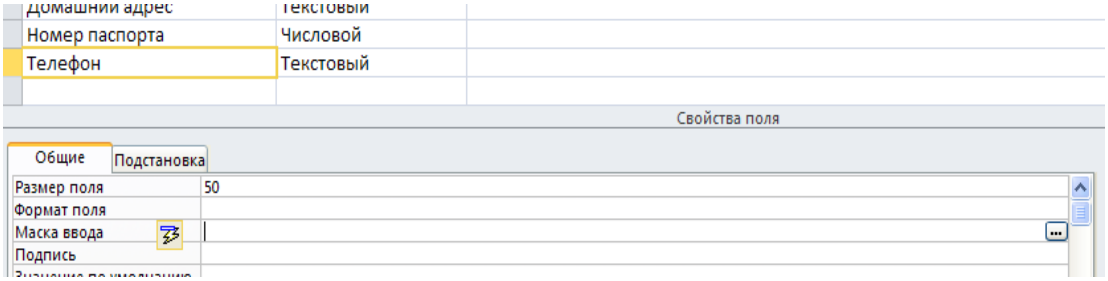


Рисунок - 3 Создание маски ввода

В окне Создание маски ввода выбираем командную кнопку Список. Введем следующие данные для создания маски как показано на рис 4. Далее выбираем командную кнопку Закрыть.

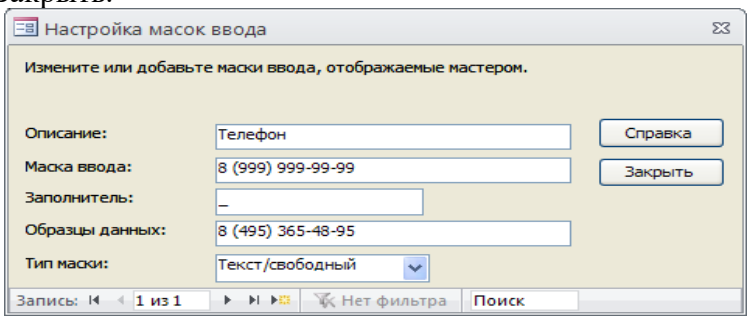


Рисунок - 4 Маска ввода для телефона

В окне Создание маски ввода выбираем созданную нами маску Телефон и нажимаем командную кнопку Далее несколько раз.

Создадим таблицу «Издательства», содержащую информацию об издателях: Наименование издательства, Город, Телефон, E-mail, Сайт издательства. Для создания ключевого поля добавим поле Код издательства. Таблица «Издательства» в режиме конструктор выглядит следующим образом:

Издательства

Имя поля	Тип данных
Код издательства	Счетчик
Издательство	Текстовый
Город	Текстовый
Телефон	Текстовый
E-mail	Текстовый
Сайт	Текстовый

Рисунок - 5 Таблица «Издательства» в режиме конструктора

Создадим таблицу «Книги», содержащую информацию о книгах: Название книги, Автор, Год издания, Число страниц, Цена. Для создания ключевого поля добавим поле Код книги, для связи с таблицей «Издательства» поле Код издательства. Таблица «Книга» в режиме конструктор выглядит следующим образом:

Книги

Имя поля	Тип данных
Код книги	Текстовый
Код издательства	Числовой
Название	Текстовый
Автор	Текстовый
Год издания	Числовой
Число страниц	Числовой
Цена (руб)	Денежный

Рисунок - 6 Таблица «Книги» в режиме конструктора

Так как данные для поля Код издательства находятся в таблице «Издательства», то ввод их нужно организовать с помощью мастера подстановок, для чего ставим курсор в поле Код издательства и в столбце «тип данных» выбираем команду мастер подстановок, откроется окно создания подстановки (Рисунок - 7). В окне создание подстановки выбираем пункт «объект «поле подстановки» получит значение из другой таблицы или другого запроса@ и нажимаем на командную кнопку далее, в открывшемся окне выбираем таблицу «Издательства» и жмем кнопку далее. Из окна доступные поля выбираем поля Код издательства и Издательство и перебрасываем в окно выбранные поля, жмем кнопку далее и готово.

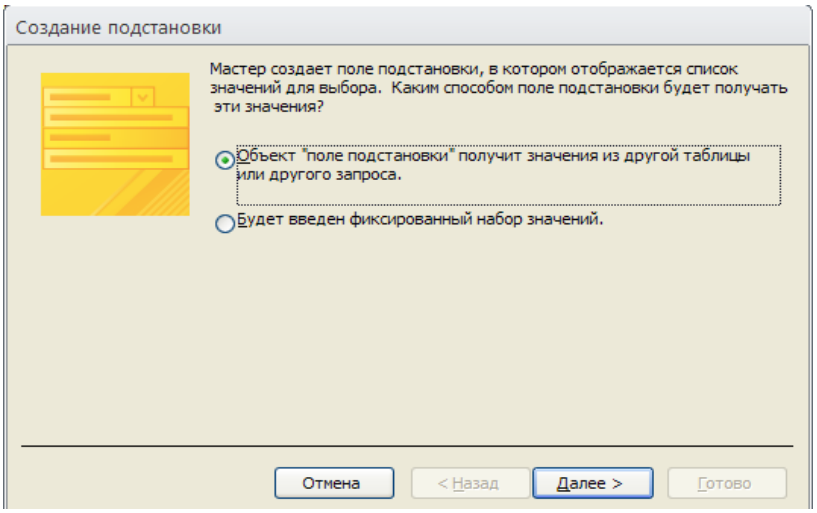


Рисунок - 7 Окно создания подстановки

Создадим таблицу «Выдача», содержащую информацию о книгах, взятых читателями в библиотеке: Дата выдачи, Дата возврата. Для создания ключевого поля добавим поле Код выдачи, для связи с таблицами

«Читатель» и «Книга» поля Код читателя и Код книги. Данные для полей Код читателя и Код книги организовать с помощью мастера подстановок.

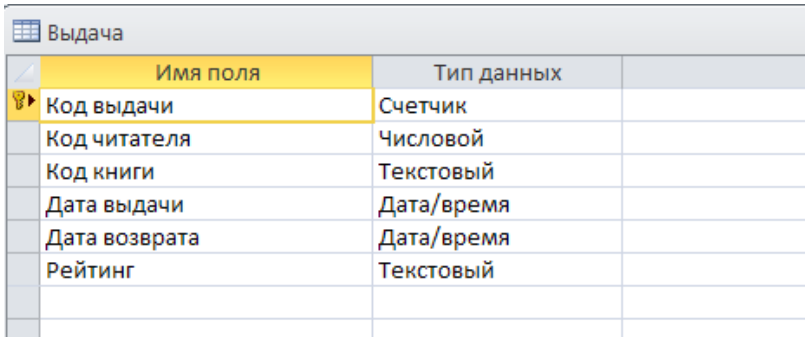


Рисунок - 8 Таблица «Выдача» в режиме конструктора

Установим связи между таблицами, для чего зададим команду Работа с базами данных / раздел Отношения/ Схема данных. Далее выбираем из раздела Связи команду Отобразить таблицу откроется окно:

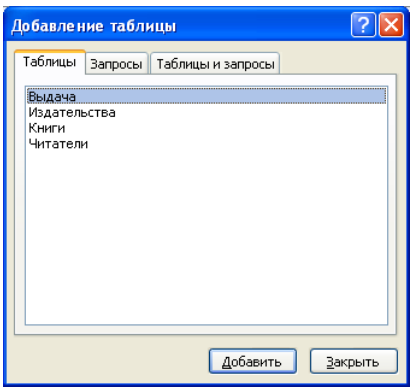


Рисунок - 9 Окно добавление таблицы

В окне «добавление таблицы» выбираем поочередно каждую таблицу и жмем командную кнопку «добавить». Устанавливаем связи между таблицами, предусмотрев обеспечение целостности данных, каскадное обновление связанных полей и каскадное удаление связанных записей, как показано на Рисунок - 10:

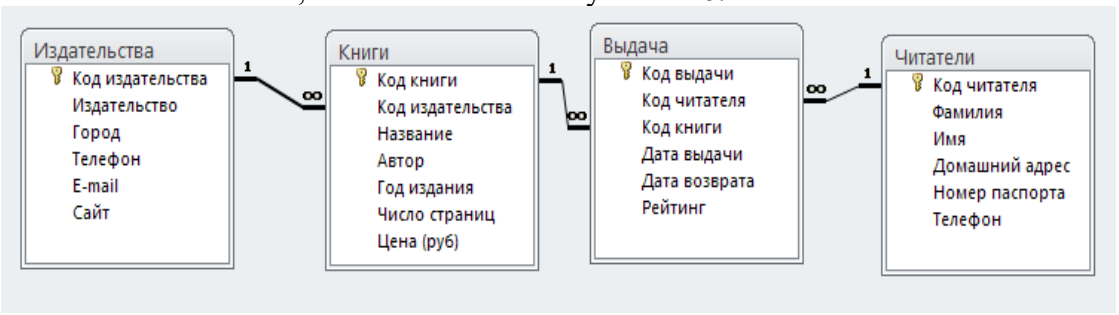


Рисунок - 10 Схема данных БД «Библиотека»

Заполняем таблицы, введя не менее 4 записей в таблицы без внешнего ключа (Таблицы «Издательства» и «Читатель») и не менее 10 записей в таблицы, содержащие поле внешнего ключа (Таблицы «Книги» и

«Выдача») как показано на следующих рисунках. Серию паспорта не заполнять, номер паспорта - 6 цифр. В телефон набрать 11-значный номер (последние цифры произвольно).

Читатели

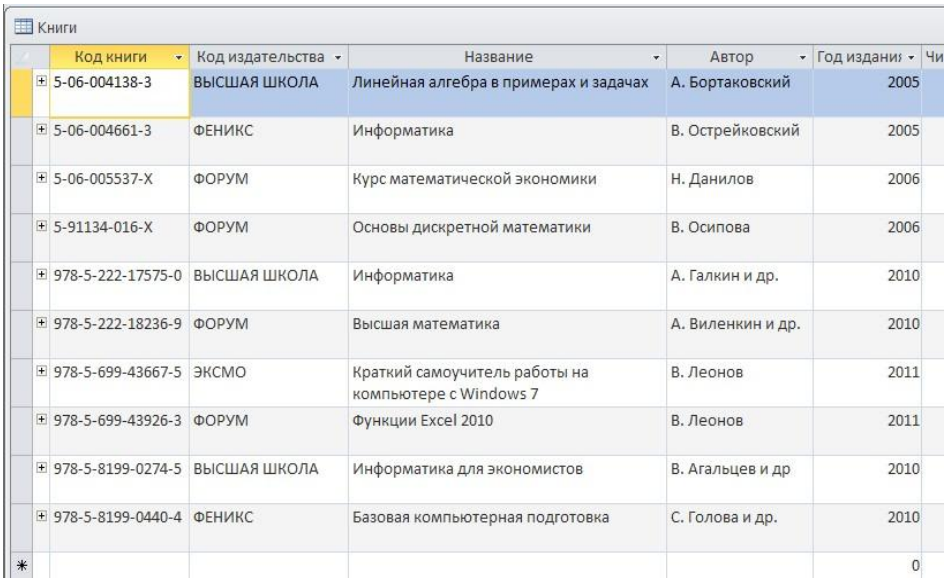
Код читате	Фамилия	Имя	Домашний адрес	Номер паспор	Телефон	Щелкн
1	Иванов	Иван	ул. Новая, 1, кв. 1	10 11 111112	8(495)-121-13-	
2	Петров	Кирилл	ул. Абромовича, 2, кв. 2	23 71 837819	8(736)-417-32-	
3	Романов	Андрей	ул. Путина, 4, кв.5	56 78 776741	8(654)-193-78-	
4	Сидоров	Виктор	ул. Гагарина, 6, кв. 56	67 97 979976	8(657)-568-90-	
5	Петрова	Юлия	ул. Космонавтов, 67, кв. 38	30 30 300307	8(007)-888-93-	
6	Жильцов	Иван	ул. Космонавтов, 7, кв. 3	30 30 367553	8(495)-778-89-	

Рисунок - 11 Таблица «Читатели»

Издательства

Код издате	Издательство	Город	Телефон	E-mail	Сайт	Щелкнит
1	ВЫСШАЯ ШКОЛА	Москва	8(495)-694-04-	secretar@vshkola.ru	vshkola.ru/p11aa1.htm	
2	ФЕНИКС	Ростов-на-Дону	8(634)-391-38-	tvoyakniga@mail.ru	phoenixrostov.ru	
3	ФОРУМ	Москва	8(495)-625-32-	forum-knigi@mail.ru	forum-books.ru	
4	ЭКСМО	Москва	8(495)-411-68-	info@eksmo.ru	eksmo.ru	
(№)						

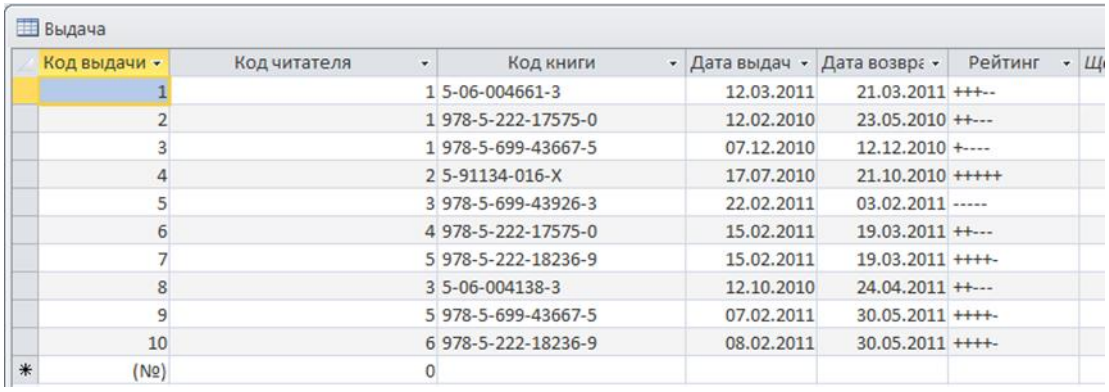
Рисунок -12 Таблица «Издательства»



Книги

Код книги	Код издательства	Название	Автор	Год издани	Чи
5-06-004138-3	ВЫСШАЯ ШКОЛА	Линейная алгебра в примерах и задачах	А. Бортаковский	2005	
5-06-004661-3	ФЕНИКС	Информатика	В. Острейковский	2005	
5-06-005537-X	ФОРУМ	Курс математической экономики	Н. Данилов	2006	
5-91134-016-X	ФОРУМ	Основы дискретной математики	В. Осипова	2006	
978-5-222-17575-0	ВЫСШАЯ ШКОЛА	Информатика	А. Галкин и др.	2010	
978-5-222-18236-9	ФОРУМ	Высшая математика	А. Виленкин и др.	2010	
978-5-699-43667-5	ЭКСМО	Краткий самоучитель работы на компьютере с Windows 7	В. Леонов	2011	
978-5-699-43926-3	ФОРУМ	Функции Excel 2010	В. Леонов	2011	
978-5-8199-0274-5	ВЫСШАЯ ШКОЛА	Информатика для экономистов	В. Агальцев и др	2010	
978-5-8199-0440-4	ФЕНИКС	Базовая компьютерная подготовка	С. Голова и др.	2010	
				0	

Рисунок - 13 Таблица «Книги»



Выдача

Код выдачи	Код читателя	Код книги	Дата выдач	Дата возвра	Рейтинг	Щ
1	1	5-06-004661-3	12.03.2011	21.03.2011	+++--	
2	1	978-5-222-17575-0	12.02.2010	23.05.2010	++---	
3	1	978-5-699-43667-5	07.12.2010	12.12.2010	+----	
4	2	5-91134-016-X	17.07.2010	21.10.2010	+++++	
5	3	978-5-699-43926-3	22.02.2011	03.02.2011	-----	
6	4	978-5-222-17575-0	15.02.2011	19.03.2011	++---	
7	5	978-5-222-18236-9	15.02.2011	19.03.2011	++++-	
8	3	5-06-004138-3	12.10.2010	24.04.2011	++---	
9	5	978-5-699-43667-5	07.02.2011	30.05.2011	++++-	
10	6	978-5-222-18236-9	08.02.2011	30.05.2011	++++-	
(№)	0					

Рисунок - 14 Таблица «Выдача»

Создать следующую форму: подчиненную форму, отображающую данные из таблиц «Издательства» и «Книги».

Для этого зададим команду Создание / раздел Формы / Мастер форм.

В появившемся окне Мастер форм из окна Таблицы и запросы сначала выбираем таблицу

«Издательства», перебрасываем все поля этой таблицы, в окно Выбранные поля, а затем выбираем таблицу «Книги» и тоже перебрасываем все поля в окно Выбранные поля.

Далее нажимаем на командную кнопку Далее и Готово.

В созданную форму добавить кнопки для перехода между записями.

Для этого зададим команду.

Открыть форму Издательства в режиме конструктора и выбрать командную кнопку

Кнопка в разделе Элементы управления.

Ставить кнопку в свободное место формы и в появившемся окне Создание кнопок выбрать команду как на Рисунок-15

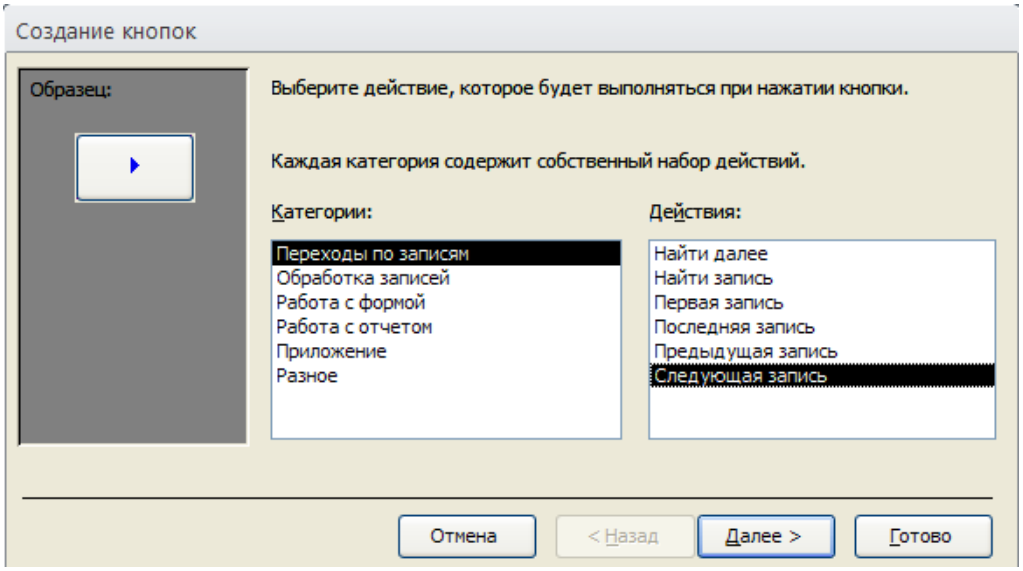


Рисунок - 15. Создание кнопок перехода между записями

Нажать кнопку Далее и выбрать текст Следующая запись. Точно так же оформить и кнопку Предыдущая запись.

Контрольные вопросы:

1. Какие базы данных назеваются реляционными?
2. Сетевые базы данных – это…
3. Распределенная база данных – это
4. Фактографическая база данных – это…
5. Какие базы данных называются документальными?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №23

Тема: Запросы к многотабличной базе данных.

Количество часов на выполнение: 4.

Цель работы: выработать практические навыки работы с базами данных, формирования запросов к базам данных, сортировке данных, созданию форм и отчетов.

Оборудование: Персональный компьютер, MS Word.

Методические указания к выполнению задания

Теоретическая часть

Запросы представляют собой средство обработки данных, хранящихся в таблицах.

Запросы осуществляют сбор запрашиваемых данных по указанным критериям из одной или нескольких таблиц, а также осуществляют поиск и структурирование данных. В сетевой технологии клиент-сервер основные базы данных хранятся на сервере баз данных, используя систему запросов можно уменьшить объем передаваемой по сети информации.

Запрос является набором условий, в соответствии с которым осуществляется выборка информации из таблиц. Запуском запроса формируется новая таблица данных, обновляющаяся при повторных запусках запроса в соответствии с изменением информации в исходных таблицах.

Существуют различные типы запросов:

- на выборку;
- запросы с параметрами;
- перекрестные;
- запросы на изменение таблиц.

Запрос с параметром представляет собой отбор записей в базовой таблице по введенным параметрам.

Запрос на выборку позволяет получить результирующую таблицу, отображающую записи из базовых таблиц, которые удовлетворяют условиям запроса.

Поле схемы запроса — верхняя часть окна конструктора запросов, куда помещаются схемы таблиц, данные из которых используются в запросе.

Бланк запроса — таблица в нижней части окна. Столбцы относятся к полям, участвующим в формировании запроса. В первой строке указываются имена всех этих полей. Вторая строка — имя таблицы, из которой извлекается соответствующее поле. Третья строка — признак сортировки. Используется лишь для ключей сортировки. Флажки в пятой строке отмечают признак вывода данного поля на экран при выполнении запроса. В следующих строках формируется условие отбора.

Формирование простых запросов

Чтобы получить определенные данные из БД пользователь использует запросы. В результате выполнения запросов образуются таблицы с временными наборами данных, записи которых включают поля из одной или нескольких таблиц.

Запросы используют подобно таблицам. Соответствующие временные наборы просматриваются в табличном представлении. На основе запросов строятся отчеты или формы. Обновленные данные во временном наборе включаются в таблицы. С помощью запросов можно по-разному выполнить доступ к одной и той же информации

Приложением MS Access обеспечивается корректная связь между таблицами БД за счет соответствующих друг другу значений эквивалентных полей данных таблиц. При включении в запросы связанных таблиц БД в окне Запрос-выборка между ними автоматически появляется соединительная линия. В случае отсутствия связи между таблицами запроса (не возникает автоматически) их можно соединить в окне Запрос-выборка. Для чего обязательно требуется наличие в них полей с совпадающими данными. Подобное соединение действует только для данного запроса, при использовании этих же таблиц в других запросах их потребуется соединить заново.

Помимо *запросов-выборок* существуют:

- *запросы на создание таблиц*, которые создают для хранения информации новую таблицу и оказываются особенно полезными при создании резервных копий;
- *запросы на добавление*, с помощью которых присоединяются записи к имеющейся таблице;
- *запросы на обновление*, позволяющие изменить значение любого поля БД для записей, которые удовлетворяют указанным критериям;

- *запросы на удаление*, с помощью которых уничтожаются в таблице все записи согласно поставленным критериям;

- *перекрёстные запросы*, упорядочивающие информацию в БД по двум и более значениям группировки и на основании общих подмножеств соответствующих групп позволяющие сделать определенные выводы.

В запросах можно осуществлять вычисления следующих типов:

- встроенные (итоговые), используемые при расчете следующих значений по группам записей или по всем записям, отобранным в результате запроса: суммы, среднего значения, числа значений, минимального или максимального значения, стандартного отклонения или дисперсии;

- пользовательские вычисления, используемые при расчетах с числовыми и строковыми значениями либо значениями дат для каждой записи с данными из одного или нескольких полей.

Вычисление итоговых значений с использованием перекрестных запросов

Перекрестные запросы применяют при расчетах и представлении данных в структуре, которая облегчает их анализ. С помощью перекрестных запросов можно подсчитать сумму, среднее, число значений или выполнить другие статистические расчеты, в итоге результаты сгруппируются в виде таблиц по 2 наборам данных: 1 - определяет заголовки столбцов, а 2 - заголовки строк. Функции, которые определяют тип статистической обработки, можно выбрать в строке Групповая операция на панели запроса.

Перекрестные запросы легко создаются в режиме Конструктора:

- создадим перекрестный запрос, используя режим Конструктора, и добавим таблицы, записи из которых необходимы для вычислений;

- добавим поля, по которым выполняются вычисления, и зададим условия отбора;

- нажмем кнопку Групповые операции на панели инструментов, в результате появится строка Групповая операция;

- в ячейках этой строки для каждого поля запроса выберем одну из статистических

функций: Sum, Avg, Min, Max, Count, StDev или Var, либо другие функции – First и Last.

Сортировка данных

Для удобства просмотра можно сортировать записи в таблице в определенной последовательности. Кнопки сортировки на панели инструментов  (или команды меню **Записи►Сортировка**) позволяют сортировать столбцы по возрастанию или по убыванию. Прежде чем щелкнуть по кнопке сортировки, следует выбрать поля, используемые для сортировки. Современные СУБД (такие как Access) никогда не сортируют таблицы физически, как это делалось раньше. Средства сортировки данных (а также фильтрации, поиска и замены) реализованы в Access как автоматически создаваемые запросы. Записи таблицы всегда располагаются в файле базы данных в том порядке, в котором они были добавлены в таблицу.

Отбор данных с помощью фильтра.

Фильтр – это набор условий, применяемых для отбора подмножества записей. В Access существуют фильтры четырех типов: фильтр по выделенному фрагменту, обычный фильтр, расширенный фильтр и фильтр по вводу.

Фильтрование данных в Access производится с помощью кнопок  либо команды меню **Записи►Фильтр**. После нажатия второй кнопки от таблицы остается одна запись. Каждое поле становится полем со списком (когда в нем находится курсор), в котором можно выбрать из списка все значения данного поля. После щелчка мышью по кнопке  выбираются записи, соответствующие измененному фильтру. Еще более сложные условия фильтрации можно задать командой меню

Записи►Фильтр►Расширенный фильтр.

Создание форм

Экранные формы в Access используются для ввода данных в базу, корректировки данных. Кроме того, они используются также для реализации просмотра баз данных по определенным условиям и для создания «заставок» и меню (так называемые несвязанные формы, т.е. формы, не связанные с какой-либо таблицей).

Access позволяет строить сложные формы с включением табличной (многострочной) части, а также множества «динамических» частей, возможно и из разных баз данных.

В экранной форме помимо информационных присутствуют и управляющие элементы.

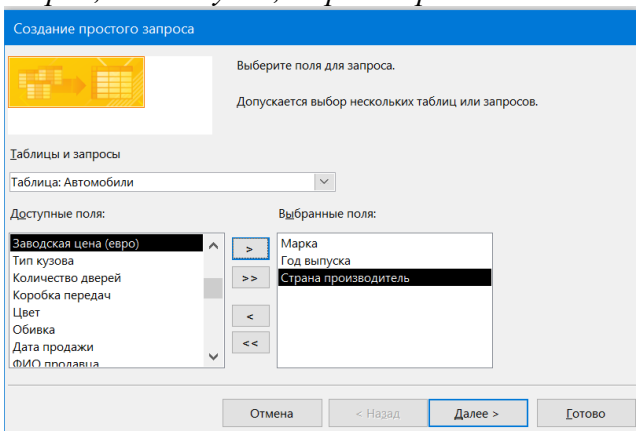
Практическая часть

Откройте базу данных «Список автомобилей».

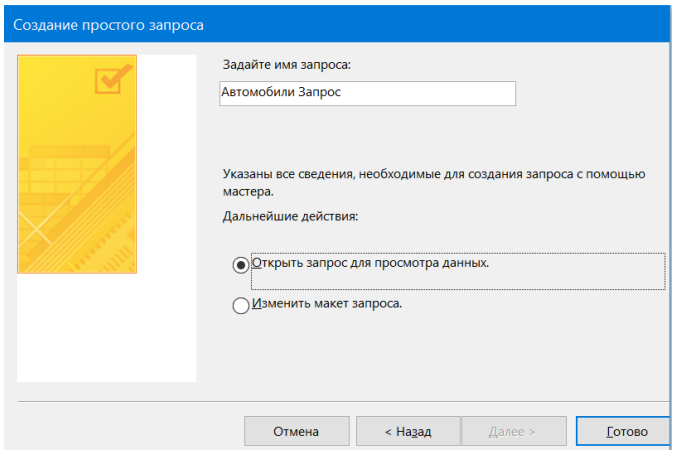
Задание 1. Простой запрос

Выполните команду: вкладка ленты Создание → Мастер запросов →Простой запрос.

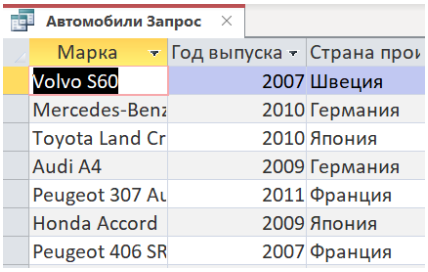
В появившемся диалоговом окне укажите таблицу *Автомобили* и выберите поля *Марка, год выпуска, страна-производитель*. Нажмите кнопку Далее.



Далее выбираем подробный запрос, вводим имя запроса и нажимаем *готово*



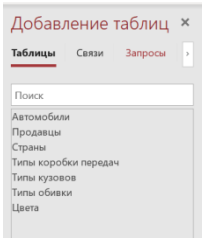
Перед вами появится запрос, в котором можно просмотреть информацию, которую запросили



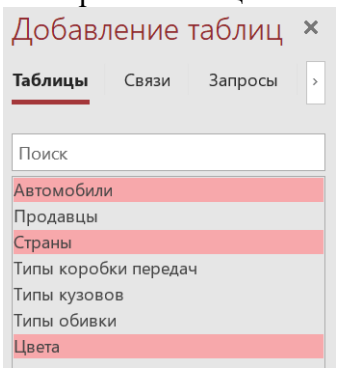
Задание 2. Запрос с помощью конструктора

Следующий запрос попробуйте создать с помощью Конструктора, для этого выполните команду: вкладка ленты Создание → Конструктор запросов.

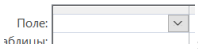
В диалоговом окне Добавление таблиц



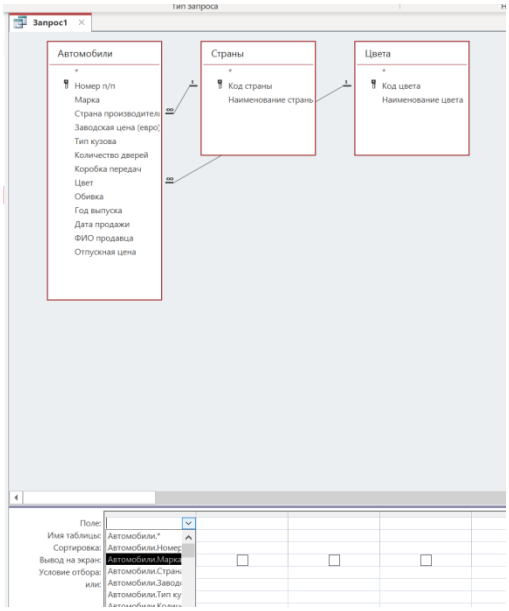
выберите таблицы



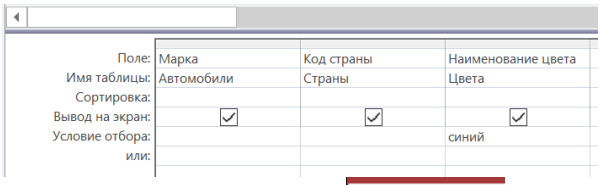
щелкните на кнопке *Добавить выбранные таблицы.*

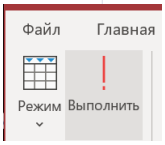
Чтобы перенести нужные поля в бланк запроса, необходимо щелкнуть левой кнопкой мыши по пустому полю, нажать на галочку 

, откуда выпадет список

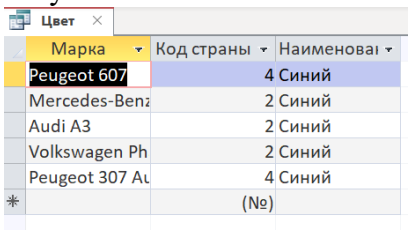


В условиях отбора написали синий цвет



Нажали кнопку выполнить. 

Результат:



Сохранить запрос (ПКМ по названию запроса, переименовать – цвет)

Задание 3. Создать запрос «Какие авто были проданы в апреле»

Понадобится таблица *автомобили. Поля «дата продажи», марка автомобилей»*

В условиях отбора, для того, чтобы выбрать апрель (04), необходимо написать конструкцию like «*.04.*»

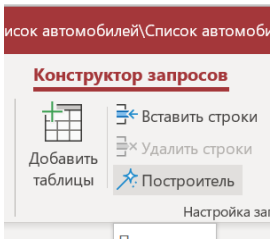
В марке автомобиля пишем «Volvo S60» и нажимаем выполнить

Задание 4. Посчитать сумму отпускной цены в рублях и подоходный налог

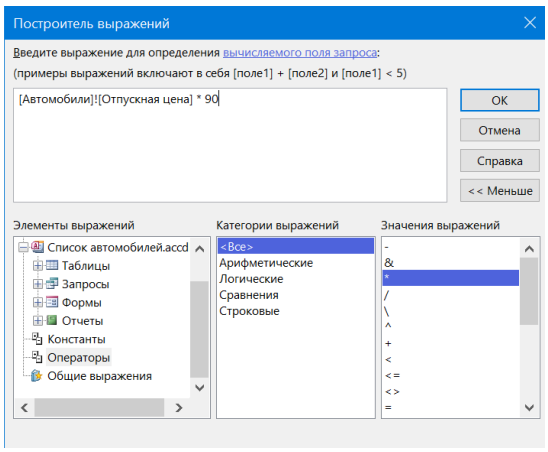
Для этого нужно воспользоваться построителем.

В конструкторе запросов выбираем таблицу «автомобили»

В пустом поле выбираем «отпускная цена». В конструкторе запросов выбираем *Построитель*



В элементах выражений выбираем с помощью двойного щелчка сначала таблицы/автомобили/отпускная цена, далее *операторы, звездочку (умножение), прописываем курс рубля (90).* Делаем как на скриншоте, если вдруг написалось что-то лишнее, то стираем ненужное.



Нажимаем Ок и выполнить

Таким же образом посчитаейте подоходный налог 13% (надо умножить на 0,13)

Задание 5. Запрос с параметром

Создайте запрос, выбрав поля: Марка, страна-производитель, цвет, год.

В поле «год», в условиях отбора создайте параметр с условием, введя [введите год]. Сделайте выборку за 2007 и 2010 год.

Задание 6. Создание формы

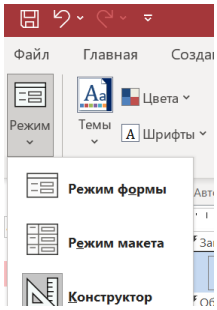
теория

Создание простой формы

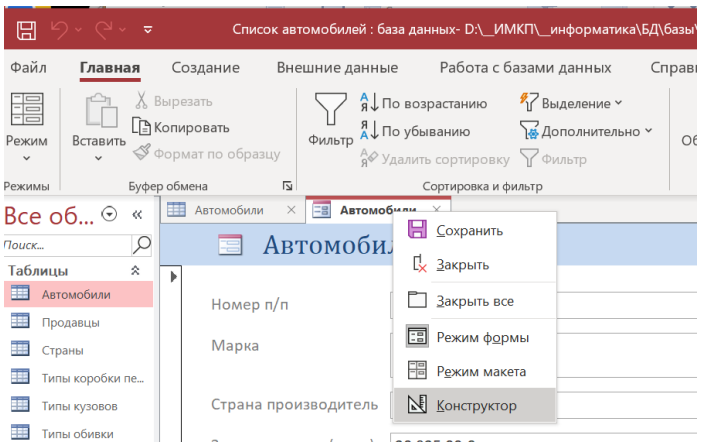
Работа с таблицей «автомобили»

Вкладка *Создание/Форма*

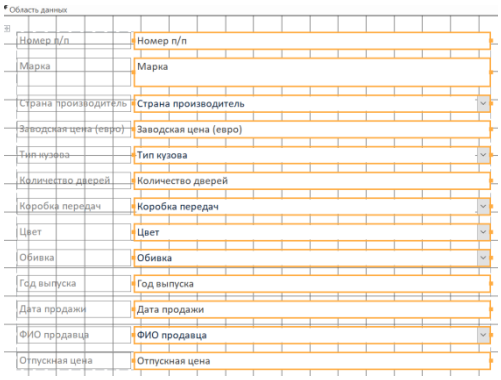
Есть три режима: Режим формы, режим макета и конструктор. Редактируем и создаем - в режиме конструктора; режим формы – просмотр, редактирование данных; в режиме макета – можно изменять размер, шрифт и цвет полей и т.д.



Правой кнопкой мыши по названию, выбираем в конекстном меню Конструктор

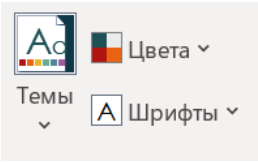


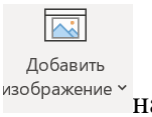
Можно изменять размер полей, выделив их и потянув за край уменьшить



Можно удалять поля, которые мы не хотим, чтобы были отражены

С помощью вкладки «тема, цвета, шрифты» можно поменять весь внешний вид формы



Можно добавить фоном картинку 

нажав правую кнопку мыши вызвать контестное меню со свойствами, где можно положение поменять, растянуть и т.д.

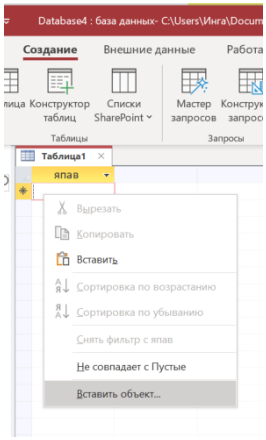
Как вставить изображение в таблицу, чтобы оно отображалось в форме

Вариант1

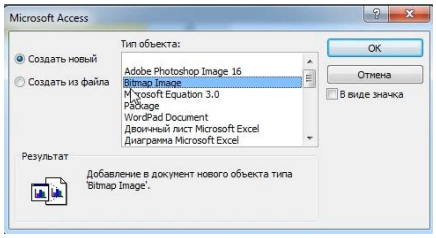
Создать в таблице столбец с типом данных *Поле объекта OLE.*

Открыть нужное изображение в программе Paint, выделить его, далее нажимаем Ctrl+C (копируем)

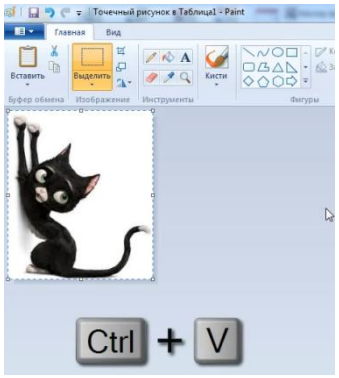
В режиме таблицы правой кнопкой мыши вызываем контестное меню *вставить объект*



Создать новый/ bitmap image



откроется программа Paint, нажимаем Ctrl+V(вставить),



закрываем Paint и таблицу,

Создаем форму, с помощью мастера форм и смотрим, что получилось

Вариант 2

Копируем изображение в интернете, ПКМ, копировать картинку

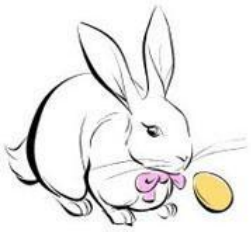
В режиме таблицы, вставляем картинку, правой кнопкой мыши *вставить*

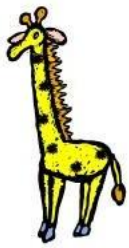
Создаем форму, смотрим, что получилось

Практика

Создайте БД «Животные»

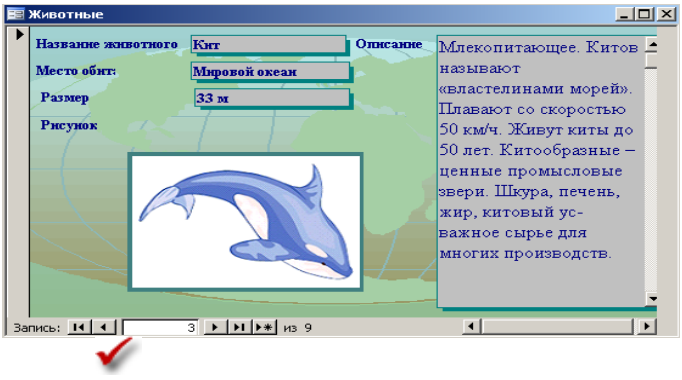
Название животного	**Место обитания**	**Размер**	**Рисунок**	**Описание**
Кит	Мировой океан	33 метра		Млекопитающее. Китов называют «властелинами морей». Плавают со скоростью 50 км/ч. Живут киты до 50 лет. Китообразные – ценные промысловые звери. Шкура, печень, жир,

				китовый ус- важное сырье для многих производств.
Черепаха	Мадагаскар	600 кг		Черепаха достаточно медлительное существо. За день проходит не более 6 километров. Питаются растениями: кактусами, листьями, травой. Очень любят воду и пьют ееочень много.
Кролик	Россия	40 см		Бегает со скоростью 70 км/ч. Селится в холмистой и песчаной местностях. Очень дружелюбные и общительные создания. Питается корой деревьев, травой.
Лягушка	Европа, Африка, Азия	12-15 см		Лягушки обитают по всему миру. Предпочитают жить в озерах, болотах, топях и трясинах. Днемони греются на солнышке. Питаются только той добычей,которую поймают.
Индийский слон	Индия	5-6 тонн		Живут слоны 60-70 лет.Считается, что слоны медлительны, но при необходимости может бежать соскоростью 30-40 км/ч. Питается молодыми побегами с высоких деревьев, травой.
Анаконда	Южная Америка	8 метров		Анаконды обычно живут в водеили около воды. Питаются небольшими птицами, млекопитающими, всевозможнойрыбой.

Попугай	Австралия, Мадагаскар	500 гр		Попугаи великолепно летают и проворно лазают по деревьям. Питаются в основном плодами, орехами, насекомыми и ягодами. Очень забавные птицы. На земле обитает 324 вида попугаев.
Белка	Европа, Россия	500 гр		Самый резвый, непоседливый зверек. Шерсть белки летом рыжеватая по цвету и походит на сосновую кору. Гнездо сооружает около вершины ели или пихты. Питаются орехами, семенами, грибами и ягодами.
Жираф	Африка	До 2 м		Жираф с виду непропорциональное животное, но если приглядеться это очень грациозное животное. Развивает скорость до 70 км/ч. Питается побегами молодых деревьев, травой.

Используйте, фон, тему, цвет, шрифт – на свой вкус.

У вас должно получиться, примерно так:



Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика
Форма контроля: дифференцированная
Ссылки на источники:*[1]*

Практическая работа №24

Тема: Выделение и обработка цифр целого числа в различных системах счисления с использованием операций целочисленной арифметики.

Количество часов на выполнение: 4.

Цель работы: сформировать умения по составлению алгоритма программы.

Оборудование: Персональный компьютер, редактор кода (любой на выбор), Python.

Методика выполнения задания
Теоретическая часть

Основные математические операторы в python:
+ — сложение
- — вычитание
* — умножение
/ — деление
% —взятие остатка от деления
** — возведение в степень
Ряд встроенных функций в Python позволяют работать с числами. В частности, функции **int()** и **float()** позволяют привести значение к типу целое и вещественное.
Модуль math – один из наиважнейших в Python. Этот модуль предоставляет обширный функционал для работы с числами.
Примеры:
math.fabs(X) - модуль X.
math.factorial(X) - факториал числа X.
math.fmod(X, Y) - остаток от деления X на Y.
math.exp(X) - e^X.
math.sqrt(X) - квадратный корень из X.
math.pi - pi = 3,1415926...
math.cos(X) - косинус X (X указывается в радианах).
math.sin(X) - синус X (X указывается в радианах).
math.tan(X) - тангенс X (X указывается в радианах). и т.п
Пример №1
Даны два целых числа x и y. Вычислить их сумму, разность, произведение и частное.
Программа:
```
# Нахождение суммы, разности и произведения двух вещественных чисел
a = float(input("Введите первое число:"))
b = float(input("Введите второе число:"))
sum = a + b
razn = a - b
proiz = a * b
cas = a / b
print('Сумма =', sum)
print('Разность =', razn)
print('Произведение =', proiz)
print('Частное =', cas)
```
Пример №2

Составить программу вычисления длины окружности, если известен радиус.

Программа

```
# Вычисления длины окружности, если известен радиус.
pi=3.14159
r = float(input("Введите радиус окружности: "))
l = 2*pi*r
print('Длина окружности =', f'{l:0.3}')
```

Пример №3

Составить программу нахождения остатка деления целочисленного числа k на n

Программа

```
# Нахождения остатка от деления целочисленного числа k на n
k = float(input("Введите число k: "))
n = float(input("Введите число n: "))
ostatok = k % n
print('Остаток от деления',ostatok)
```

Пример №4

Найдите значения вычисления выражений $z=\sin(x+\pi*(y/2))$

Программа

```
# Значения вычисления выражений z и y
import math
pi = 3.14
x = float(input("Введите число x: "))
y = math.sqrt(x)
z = math.sin(x + pi*y / 2)
print("Значение выражения y = ",y)
print("Значение выражения z = ",f"{z:0.2}")
```

Представление числа

При обычном определении числовой переменной она получает значение в десятичной системе. Но кроме десятичной в Python мы можем использовать двоичную, восьмеричную и шестнадцатеричную системы.

Для определения числа в двоичной системе перед его значением ставится 0 и префикс **b**:

```
1    x = 0b101 # 101 в двоичной системе равно 5
```

Для определения числа в восьмеричной системе перед его значением ставится 0 и префикс **o**:

```
1    a = 0o11 # 11 в восьмеричной системе равно 9
```

Для определения числа в шестнадцатеричной системе перед его значением ставится 0 и префикс **x**:

```
1    y = 0x0a # a в шестнадцатеричной системе равно 10
```

И с числами в других системах измерения также можно проводить арифметические операции:

```
1    x = 0b101 # 5
2    y = 0x0a # 10
3    z = x + y # 15
4    print("{0} in binary {0:08b}   in hex {0:02x} in octal {0:02o}".format(z))
```

Для вывода числа в различных системах исчисления используются функция format, которая вызывается у строки. В эту строку передаются различные форматы.

Для двоичной системы "{0:08b}", где число 8 указывает, сколько знаков должно быть в

записи числа. Если знаков указано больше, чем требуется для числа, то ненужные позиции заполняются нулями.
Для шестнадцатеричной системы применяется формат "{0:02x}". И здесь все аналогично - запись числа состоит из двух знаков, если один знак не нужен, то вместо него вставляется ноль. А для записи в восьмеричной системе используется формат "{0:02o}".

Практическая часть

В системе программирования **Python** составьте программу по заданию. Протестируйте работу программы. Выполните задания на модификацию созданной программы.
Методические указания по выполнению задания:
1. а) В среде **Python** открыть новый файл для создания программы командой **Ctrl** + **N**
б) Набрать в **редакторе** системы **Python** следующую программу:

```
a = int(input("Введите a"))
b = int(input("Введите b"))
c = int(input("Введите c"))
S=a*b*c
print(S)
```

в) Запустить данную программу на выполнение и проверить правильность её работы для чисел 2, 4 и 6.
г) Запустить данную программу на выполнение и проверить правильность её работы для чисел 1, 0 и -1.
д) Запустить данную программу на выполнение и проверить правильность её работы для чисел -2, 3 и 10.

2. Написать программу, которая присваивает целой переменной A значение 10 и выводит это значение на экран.

3. Написать программу, которая запрашивает ввод целого числа в переменную B и выводит это число на экран. Проверить правильность работы программы на числах 1, -5, 256, 10455.

4. Написать программу, которая запрашивает ввод вещественного числа в переменную C, умножает это число на 2 и выводит результат на экран. Проверить правильность работы программы на числах 2.5, -7.33, 0, 782.234.

5. Написать программу для ввода значения величины X целого типа, присваивания величине Y действительного типа значения 5.5, вычисления значения величины Z = X - Y и вывода значения величины Z. Протестировать программу для X=5.5, X=0, X=-10.2

6. Написать программу для ввода значения величины X целого типа, присваивания величине Y действительного типа значения 2.5 , вычисления значения величины Z=X/Y и вывода значения величины Z. Протестировать программу для X=5, X=0, X=-8.75

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.
3.Результаты выполнения заданий.
4. Ответы на контрольные вопросы.
Практическая работа сохраняется в личной папке на сетевом диске.
Затем прикрепляется в Электронный курс https://el.istu.edu Информатика
Форма контроля: дифференцированная
Ссылки на источники:*[1]*

Практическая работа № 25

Тема: Решение задач методом перебора
Количество часов на выполнение: 4.
Цель работы: Изучить решение задач методом перебора при помощи средств языка Python
Оборудование: Персональный компьютер, MS Word, редактор кода, Python.

Методика выполнения задания

Теоретическая часть

Метод перебора (метод равномерного поиска, перебор по сетке) — простейший из методов поиска значений действительно-значных функций по какому-либо из критериев сравнения (на максимум, на минимум, на определённую константу). Применительно к экстремальным задачам является примером прямого метода условной одномерной пассивной оптимизации.

Решение уравнения перебором в целых числах в Python

Приведем пример программы на использование цикла for и функций в программах Python для поиска решений уравнения в целых числах В программе в функции задаётся уравнение, которое нужно решить в целых числах. Пользователем задаётся диапазон, в котором ищутся решения [lowerBound, upperBound], и свободный член уравнения a. Компьютер перебирает все числа для всех переменных в заданном диапазоне , проверяет являются ли они решением уравнения и выводит найденные решения уравнения на экран.

Для примера мы разберём поиск решений для уравнения x^2 + y^2 + z^2 = a. Вы можете легко поменять уравнение в функции eq()

Возведение в степень в Python осуществляется с помощью **, тогда выражение для нашего уравнения запишется в виде

x ** 2 + y ** 2 + z ** 2

Уравнение eq(x, y, z) объявляется с помощью функции def eq(x, y, z). def это ключевое слово в python для задания процедур или функций, eq это название функции, задающей выражение левой части уравнения в программе, а x, y, z это аргументы уравнения. С помощью команды return в конце функции в python обозначается то, что возвращает функция при ее вызове в программе, после return мы и записываем выражение левой части уравнения. Подробнее о функциях в Python

С помощью выражения int(input()) вводится в Python вводится целочисленная информация с клавиатуры, значения границ диапазона и свободный член уравнения. Подробнее об вводе данных с клавиатуры в Python

lowerBound = int(input("Введите нижнюю границу поиска. "))
upperBound = int(input("Введите верхнюю границу поиска. "))
a = int(input("Введите свободный член x^2 + y^2 + z^2. "))

Для перебора всех переменных x, y и z используются вложенные циклы for. Подробнее об циклах for в python. Все аргументы перебираются в заданном диапазоне и каждый раз проверяется, равно ли уравнение с подставленными значениями переменных свободному члену. Если это условие выполняется, то на экран выводится

решение уравнения

Полная программа на Python для решения уравнения в целых числах методом перебора

```
def eq(x, y, z):
  return x ** 2 + y ** 2 + z ** 2
lowerBound = int(input("Введите нижнюю границу поиска. "))
upperBound = int(input("Введите верхнюю границу поиска. "))
a = int(input("Введите свободный член x^2 + y^2 + z^2. "))
for x in range(lowerBound, upperBound):
  for y in range(lowerBound, upperBound):
    for z in range(lowerBound, upperBound):
      if (eq(x, y, z) == a):
        print("x=",x, end = ", ")
        print("y=", y, end = ", ")
        print("z=",z, end = " ")
```

Практическая часть

Задание 1. 2*x-1=sin(x)

Напишите программу, которая находит все решения заданного вам уравнения на интервале [-5;5]. Программа должна выполнить следующие действия:

1. Определяет и выводит на экран интервалы, на которых расположены корни уравнения.
2. На каждом интервале, используя метод перебора, ищет решение с точностью 0,001 и выводит полученные решения на экран.

Вычисление функции оформите в виде подпрограммы.

Задание 2. Напишите программу, которая находит все решения заданного вам уравнения на интервале [-5;5]. Программа должна выполнить следующие действия:

1. Определяет и выводит на экран интервалы, на которых расположены корни уравнения.
2. На каждом интервале, используя метод перебора, ищет решение с точностью 0,001 и выводит полученные решения на экран.

Вычисление функции, стоящей в левой части уравнения, оформите в виде подпрограммы.

Уровень А. Интервалы, на которых расположены корни, можно найти с помощью электронных таблиц. Программа запрашивает левую границу очередного интервала и выводит найденный корень уравнения.

Пример:

Введите левую границу интервала:

1.5

Решение: 1.7201

Уровень В. Составить две программы, одна из которых выделяет все интервалы, на которых находятся корни, а вторая запрашивает левую границу очередного интервала и выводит найденный корень уравнения.

Уровень С. Составить одну программу, которая работает полностью автоматически: достаточно ввести в программу функцию и запустить. Программа находит все интервалы, на которых расположены корни и уточняет решения. После того, как очередной интервал найден, программа выводит его на экран и, применяя метод перебора с нужным шагом, уточняет решение.

Пример:

Интервал [-2;-1,5]

Решение: -1.7201

Интервал [1,5;2]

Решение: 1.7201

Варианты заданий по теме «Решение уравнений»

№	Уравнение	Ответы
1	$x^3-8x+1=5\sin x$	-2.945 0.077 2.854
2	$x^3-8x+1=-2\sin x-10\cos x$	-2.233 1.088 3.286
3	$x^3-8x+1=-12\sin x-10\cos x$	-1.332 1.975 3.379
4	$x^3-8x+1=5\sin x+12\cos x$	-3.377 -0.683 2.347
5	$x^3-8x+1=-5\sin x+12\cos x$	-3.468 -1.210 1.798
6	$x^3+2x^2-8x+1=-2\sin x-15\cos x$	-3.123 1.493 2.638
7	$x^3+2x^2-8x+1=-2\sin x+5\cos x$	-4.200 -0.497 1.639
8	$x^3+2x^2-8x+1=-12\sin x+8\cos x$	-4.495 -1.386 0.850
9	$x^3+2x^2-8x+1=5\sin x+12\cos x$	-4.133 -0.638 1.941
10	$x^3+2x^2-8x+1=-5\sin x+12\cos x$	-4.356 -1.042 1.440
11	$x^3+3x^2-9x-5=-\cos x$	-4.690 -0.406 2.212
12	$x^3+3x^2-9x-20=-8\sin x+8\cos x$	-4.488 -2.100 2.363
13	$x^3+3x^2-9x-5=-5\sin x+5\cos x$	-4.834 -1.066 1.776
14	$x^3+3x^2-9x-5=-10\sin x+10\cos x$	-4.931 -1.441 1.448
15	$x^3+3x^2-9x-20=-18\cos x+25\cos x$	-4.756 -2.101 1.720
16	$x^3+2x^2-6x-5=-\sin x-\cos x$	-3.285 -0.707 2.039
17	$x^3+2x^2-6x-5=-\sin x-15\cos x$	-1.986 1.086 2.704
18	$x^3+2x^2-6x-5=-10\sin x-20\cos x$	-1.281 1.973 2.772
19	$x^3+2x^2-6x-5=-10\sin x+10\cos x$	-4.040 -1.866 1.316
20	$x^3+2x^2-6x-5=-18\sin x+25\cos x$	-4.395 -1.968 1.196

Контрольные вопросы:

1. Как работает цикл for в python?
2. Какая функция выводит что-либо в консоль (на экран монитора)?
3. Как вести данные с клавиатуры в python?
4. Какие существуют типы переменных у чисел в python?
5. Переменная int это?
6. Имена переменных не могут включат в себя:

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика
Форма контроля: дифференцированная
Ссылки на источники:*[1]*

Практическая работа № 26

Тема: Обработка данных, хранящихся в файлах.
Количество часов на выполнение: 2.
Цель работы: научиться обрабатывать данные, хранящиеся в файлах в Python.
Оборудование: Персональный компьютер, MS Word, редактор кода, Python.

Методика выполнения задания

Теоретическая часть.

Файл — это всего лишь набор данных, сохраненный в виде последовательности битов на компьютере. Информация хранится в куче данных (структура данных) и имеет название «имя файла» (filename).

В Python существует два типа файлов:
Текстовые
Бинарные
Текстовые файлы

Это файлы с человекочитаемым содержимым. В них хранятся последовательности символов, которые понимает человек. Блокнот и другие стандартные редакторы умеют читать и редактировать этот тип файлов.

Текст может храниться в двух форматах: (.txt) — простой текст и (.rtf) — «формат обогащенного текста».

Бинарные файлы

В бинарных файлах данные отображаются в закодированной форме (с использованием только нулей (0) и единиц (1) вместо простых символов). В большинстве случаев это просто последовательности битов.

Они хранятся в формате .bin.

Любую операцию с файлом можно разбить на три крупных этапа:
Открытие файла
Выполнение операции (запись, чтение)
Закрытие файла
Открытие файла
Метод open()

В Python есть встроенная функция open(). С ее помощью можно открыть любой файл на компьютере. Технически Python создает на его основе объект.

Синтаксис следующий:
f = open(file_name, access_mode)
Где,
file_name = имя открываемого файла
access_mode = режим открытия файла.

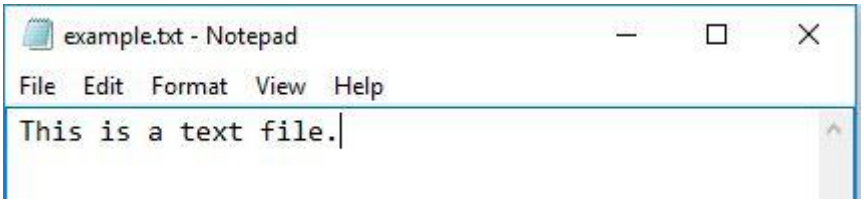
Он может быть: для чтения, записи и т. д. По умолчанию используется режим чтения (r), если другое не указано. Далее полный список режимов открытия файла

Режим	Описание
r	Только для чтения.
w	Только для записи. Создаст новый файл, если не найдет с указанным именем.
rb	Только для чтения (бинарный).

wb	Только для записи (бинарный). Создаст новый файл, если не найдет с указанным именем.
r+	Для чтения и записи.
rb+	Для чтения и записи (бинарный).
w+	Для чтения и записи. Создаст новый файл для записи, если не найдет с указанным именем.
wb+	Для чтения и записи (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.
a	Откроет для добавления нового содержимого. Создаст новый файл для записи, если не найдет с указанным именем.
a+	Откроет для добавления нового содержимого. Создаст новый файл для чтения записи, если не найдет с указанным именем.
ab	Откроет для добавления нового содержимого (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.
ab+	Откроет для добавления нового содержимого (бинарный). Создаст новый файл для чтения записи, если не найдет с указанным именем.

Пример

Создадим текстовый файл example.txt и сохраним его в рабочей директории.



Следующий код используется для его открытия.

```
f = open('example.txt','r')  # открыть файл из рабочей директории в реж
fp = open('C:/xyz.txt','r')  # открыть файл из любого каталога
```

В этом примере f — переменная-указатель на файл example.txt.

Следующий код используется для вывода содержимого файла и информации о нем.

Стоит обратить внимание, что в Windows стандартной кодировкой является cp1252,

```
>>> print(*f) # выводим содержимое файла
This is a text file.
>>> print(f) # выводим объект
<_io.TextIOWrapper name='example.txt' mode='r' encoding='cp1252'>
```

a в Linux — utf-08.

Закрытие файла

Метод close()

После открытия файла в Python его нужно закрыть. Таким образом освобождаются ресурсы и убирается мусор. Python автоматически закрывает файл, когда объект присваивается другому файлу.

Существуют следующие способы:

Способ №1

Проще всего после открытия файла закрыть его, используя метод close().

```
f = open('example.txt','r')
# работа с файлом
f.close()
```

После закрытия этот файл нельзя будет использовать до тех пор, пока заново его не открыть.

Способ №2

Также можно написать try/finally, которое гарантирует, что если после открытия файла операции с ним приводят к исключениям, он закроется автоматически.

Без него программа завершается некорректно.

Вот как сделать это исключение:

```
f = open('example.txt','r')
try:
   # работа с файлом
finally:
   f.close()
```

Файл нужно открыть до инструкции try, потому что если инструкция open сама по себе вызовет ошибку, то файл не будет открываться для последующего закрытия.

Этот метод гарантирует, что если операции над файлом вызовут исключения, то он закроется до того как программа остановится.

Способ №3

Инструкция with

Еще один подход — использовать инструкцию with, которая упрощает обработку исключений с помощью инкапсуляции начальных операций, а также задач по закрытию и очистке.

В таком случае инструкция close не нужна, потому что with автоматически закроет файл.

Вот как это реализовать в коде.

```
with open('example.txt') as f:
    # работа с файлом
```

Чтение и запись файлов в Python

В Python файлы можно читать или записывать информацию в них с помощью соответствующих режимов.

Функция read()

Функция read() используется для чтения содержимого файла после открытия его в режиме чтения (r).

Синтаксис

```
file.read(size)
```

Где,

file = объект файла

size = количество символов, которые нужно прочитать. Если не указать, то файл прочитается целиком.

Пример

```
>>> f = open('example.txt','r')
>>> f.read(7)  # чтение 7 символов из example.txt
'This is '
```

Интерпретатор прочитал 7 символов файла и если снова использовать функцию read(), то чтение начнется с 8-го символа.

```
>>> f.read(7)  # чтение следующих 7 символов
' a text'
```

Функция readline()

Функция readline() используется для построчного чтения содержимого файла. Она используется для крупных файлов. С ее помощью можно получать доступ к любой строке в любой момент.

Пример

Создадим файл test.txt с нескольким строками:

```
This is line1.
This is line2.
This is line3.
```

Посмотрим, как функция readline() работает в test.txt.

```
>>> x = open('test.txt','r')
>>> x.readline()  # прочитать первую строку
This is line1.
>>> x.readline(2)  # прочитать вторую строку
This is line2.
>>> x.readlines()  # прочитать все строки
['This is line1.','This is line2.','This is line3.']
```

Обратите внимание, как в последнем случае строки отделены друг от друга.

Функция write()

Функция write() используется для записи в файлы Python, открытые в режиме записи.

Если пытаться открыть файл, которого не существует, в этом режиме, тогда будет создан новый.

Синтаксис

```
file.write(string)
```

Пример

Предположим, файла xyz.txt не существует. Он будет создан при попытке открыть его в режиме чтения.

```
>>> f = open('xyz.txt','w')  # открытие в режиме записи
>>> f.write('Hello \n World')  # запись Hello World в файл
Hello
World
>>> f.close()  # закрытие файла
```

Переименование файлов в Python
Функция rename()
Функция rename() используется для переименовывания файлов в Python. Для ее использования сперва нужно импортировать модуль os.
Синтаксис следующий.

```
import os
os.rename(src,dest)
```

Где,
src = файл, который нужно переименовать
dest = новое имя файла

Пример

```
>>> import os
>>> # переименование xyz.txt в abc.txt
>>> os.rename("xyz.txt","abc.txt")
```

Текущая позиция в файлах Python

В Python возможно узнать текущую позицию в файле с помощью функции tell(). Таким же образом можно изменить текущую позицию командой seek().

Пример

```
>>> f = open('example.txt')  # example.txt, который мы создали ранее
>>> f.read(4)  # давайте сначала перейдем к 4-й позиции
This
>>> f.tell()  # возвращает текущую позицию
4
>>> f.seek(0,0)  # вернем положение на 0 снова
```

Методы файла в Python

file.close()	закрывает открытый файл
file.fileno()	возвращает целочисленный дескриптор файла
file.flush()	очищает внутренний буфер
file.isatty()	возвращает True, если файл привязан к

	терминалу
file.next()	возвращает следующую строку файла
file.read(n)	чтение первых n символов файла
file.readline()	читает одну строчку строки или файла
file.readlines()	читает и возвращает список всех строк в файле
file.seek(offset[,whene])	устанавливает текущую позицию в файле
file.seekable()	проверяет, поддерживает ли файл случайный доступ. Возвращает True, если да
file.tell()	возвращает текущую позицию в файле
file.truncate(n)	уменьшает размер файл. Если n указала, то файл обрезается до n байт, если нет — до текущей позиции
file.write(str)	добавляет строку str в файл
file.writelines(sequence)	добавляет последовательность строк в файл

Практическая часть

Задание 1. Написана программа для обработки данных из файла input.txt

```
n=int(f.readline())
s=0; min_dif=10001
for i in range(n):
    a,b=map(int,f.readline().split())
    s+=max(a,b)
    dif=abs(a-b)
    if dif%5!=0 and dif<min_dif:
        min_dif=dif
if s%5!=0:
    print(s)
else:
    print(s-min_dif)
```

Организуйте ввод из файла и запишите полученный результат

Задание 2. Написать программу «Совет на день» и протестировать ее на файле input.txt. Определить какой строки не хватает

```
def f(ln,le,file):
    with open(file)as t:
        fl=t.readlines()[ln:le]
        for i in fl:
            print( i.strip())

from random import randint

sum_st=sum(1 for n in open('input.txt'))
ln,le=randint(1,sum_st),randint(1,sum_st)
_____
    ln,le=le,ln
if ln==le or le==ln+1:
    print('попробуй еще раз' )
else:
    f(ln,le,'input.txt')
```

Задание 3.

Имеется текстовый файл **prices.txt** с информацией о заказе из интернет-магазина. В нем каждая строка с помощью символа табуляции \t разделена на три колонки:

наименование товара;

количество товара (целое число);

цена (в рублях) товара за 1 шт. (целое число).

Напишите программу, подсчитывающую общую стоимость заказа.

Контрольные вопросы:

1. Какие типы файлов могут обрабатываться в Python?
2. Какой метод позволяет записать данные в файл?
3. Какие бывают способы чтения и записи файлов Python?
4. Какой метод используется для записи в файл в Python?
5. Сколько всего типов данных в Python?
6. Какое расширение имеют файлы Пайтон?
7. Какие бывают типы данных в Python?
8. Как сохранить данные в файл Python?
9. Какая функция открывает файл в Python?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 27

Тема: Использование подпрограмм стандартной библиотеки языка программирования.

Количество часов на выполнение: 4

Цель работы:

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Подпрограмма – это отдельная часть программы, имеющая имя и решающая свою отдельную задачу. Располагается подпрограмма в начале основной программы и может быть запущена (вызвана) из основной программы по указанию имени.

Использование подпрограмм позволяет избежать дублирования кода, в случае если необходимо один и тот же код писать в разных местах программы.

Библиотеки, которые импортируются в программу (например, математическая биб лиотека math) состоят из подпрограмм, которые уже кем-то составлены. Программистам не нужно думать о том, какие алгоритмы в них реализованы, а просто применяют их, задумываясь только о том, что именно они делают. Это большая экономия времени. Нет нужды писать алгоритм, который уже был кем-то написан.

Каждая подпрограмма должна решать только одну задачу, либо только что-то вы числять, либо выводить какие-либо данные, либо делать что-то еще. Подпрограммы бывают двух типов - процедуры и функции. Подпрограммы-процедуры выполняют некоторые действия, например, выводят ре зультат на экран в определенном виде (простой пример, оператор print() - это стандартная

подпрограмма-процедура, которая выводит данные на экран).

Подпрограммы-функции возвращают результат (число, символьную строчку и т.д.),

который мы можем использовать в основной программе.

Давайте попробуем написать простую процедуру:

Предположим, что нам нужно выводить на экран строку "Error" каждый раз, когда в

коде может возникнуть ошибка по вине пользователя Например, когда он вводит неверные данные.

Это можно сделать, написав оператор

print("Error")

А теперь представим, что такую строчку нужно вставить во многих местах программы. Конечно, можно везде ее просто написать. Но это решение имеет два недостатка.

1) Данная строка будет храниться в памяти много раз.

2) Если мы захотим изменить вывод при ошибке, то придется менять эту строку по всей программе, что достаточно неудобно

Для таких случаев процедуры и нужны.

Программа с процедурой может выглядеть следующим образом:

```
def printError():  # описание процедуры
    print("Error")

...
printError()      # запуск процедуры на выполнение.
                  # Просто указываем имя процедуры, которую хотим выполнить
...
printError()
```

Внимание!

1. Процедура начинается со слова `def` (от англ. - `define` - определить). После имени процедуры записаны пустые скобки и двоеточие. Внутри скобок можно указывать параметры (об этом будем говорить позднее).
2. Все операторы, которые выполняются в процедуре, записываются с отступом.
3. Чтобы выполнить процедуру, в основной программе необходимо вызвать ее **по имени и не забыть написать скобки**!
4. Вызывать процедуру в программе можно сколько угодно раз.

Параметры и аргументы

А теперь представим, что нам необходимо в ответ на ошибку пользователя вывести разные сообщения, в зависимости от того, какую именно ошибку он сделал. В этом случае можно для каждой ошибки написать свою процедуру:

```
def printErrorZero():
    print("Error. Division by zero!")
def printErrorInput():
    print("Error in input!")
```

А если возможных ошибок будет намного больше? Такое решение нам не подойдет!

Надо научиться управлять процедурой, указывая ей, какое сообщение на ошибку нужно

вывести.
Для этого нам понадобятся параметры, которые мы будем записывать в круглых скобках, после имени процедуры

```
def printError(s):
    print("s")
```

В данной процедуре **s - это параметр** - специальная переменная, которая позволяет управлять процедурой.

Задача

Оформите процедуру, которая выводит на экран фразу "Error. Division by zero!". Дайте правильное название процедуре.

Запрещенные операторы: return
Вставьте недостающие фрагменты кода

```
# Процедура, которая выводит на экран
# фразу "Error. Division by zero!"
Поле для ответа
n = int(input())
if n == 0:
    printError()
else:
    print(5//n)
```

Задача
В программе необходимо добавить вызовы процедуры таким образом, чтобы при вводе значения 0 выводилась на экран ошибка "Error: division by zero!". А при вводе четного числа выводилась ошибка "Error in input!". Ваша задача - оформить правильный вызов процедуры.

Запрещенные операторы: return;print
Вставьте недостающие фрагменты кода

```
import sys
def outputError(s):
    sys.stdout.write(s)
n = int(input())
if n == 0:
    #Error: division by zero!
elif n % 2 == 0:
   #Error in input!
```

Задача
Напишите процедуру, которая принимает параметр - натуральное число N - и выводит на

экран две линии из N символов.

Пример:

Входные данные

7

Выходные данные

Запрещенные операторы: return;exit
Вставьте недостающие фрагменты кода:

Напишите программу ниже

```
1 Поле для ответа
N = int(input())
printLine(N)
```

ПРЯМОУГОЛЬНИК

Задача

Напишите процедуру, которая принимает один параметр – натуральное число N, – и выводит на экран прямоугольник длиной N и высотой 3 символа (в качестве символа используйте английскую букву 'o').

Пример:

Входные данные

7

Выходные данные

ooooooo

ooooooo

ooooooo

Запрещенные операторы: return;exit
Вставьте недостающие фрагменты кода
Напишите программу ниже

```
1 Поле для ответа
N = int(input())
printRectangle(N)
```

ДВА ПАРАМЕТРА

Задача

Напишите процедуру, которая принимает два параметра – натуральное число N, и строку S – и выводит на экран строку S, повторив ее N раз.

Примеры

№	Входные данные	Выходные данные
1	3	okokok

№	Входные данные	Выходные данные

ok

Вставьте недостающие фрагменты кода

Напишите программу ниже:

```
# процедура
1 Поле для ответа
# основная программа
n = int(input())
s = input()
printLine(n, s)
```

1. Составить подпрограмму вычисления площади выпуклого четырехугольника ,заданного длинами своих сторон и одной из диагоналей. Решить задачу двумя способами: в первом случае составить подпрограмму – функцию.
2. Составить функцию логического типа, определяющую является ли слово палиндромом.
3. Составить процедуру обнуления всех положительных элементов в одномерном массиве A. С помощью этой процедуры обнулить все положительные элементы в *k*-м столбце произвольной матрицы.
4. Вычислить сумму n членов последовательности x0=1, x1=1,…,xk=0,7xk-1+1,1xk-2, k=2,3… При решении задачи использовать рекурсию.

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №

Количество часов на выполнение: 2, из них на практическую подготовку - 2ч. (при наличии).

Цель работы:

Оборудование: Персональный компьютер

Задание:

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №

Количество часов на выполнение: 2, из них на практическую подготовку - 2ч. (при наличии).

Цель работы:

Оборудование: Персональный компьютер

Задание:

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 28

Тема: Разработка подпрограмм.

Количество часов на выполнение: 4

Цель работы: изучение подпрограмм в Python, применение на практике.

Оборудование: Персональный компьютер, Python, любой редактор разработки (Sublime text, VS code, Notepad, Pycharm).

Методика выполнения задания
Теоретическая часть

Подпрограммами в программировании называют вспомогательные алгоритмы.

Они необходимы в ситуации, когда в разных частях программы необходимо выполнять одни и те же действия несколько раз. В таком случае повторяемые операторы оформляются в виде функции или процедуры, к которой можно обращаться и вызывать ее выполнение из разных частей программы.
В python существует два вида реализации подпрограмм: функции в роли процедуры и функции в классическом понимании.
Функция в роли процедуры призвана не возвратить значение в основную программу, а вывести его, либо выполнить какие-либо действия с глобальными переменными, при этом не возвращая полученные значения основной программе (не используя ключевое слово `return`).
Процедура – это подпрограмма, которая выполняет некоторые действия после вызова её из основной программы или другой процедуры. Каждая процедура имеет уникальное *имя,* может иметь произвольное количество входных *параметров*. При вызове процедуры указываются *фактические значения параметров*.
Локальные переменные **–** это переменные, определённые в процедуре, они доступны только внутри процедуры.
Глобальные переменные **–** это переменные, определённые в основной программе. Они доступны внутри процедуры только для чтения, а для изменения требуется объявить их в процедуре после служебного слова global.

```
x = 3 # глобальная переменная
def pr(a): # процедура с параметром
  a = 4 # локальная переменная
  print (a) # 4
pr(x) # передача параметра глобальной переменной (3)
```

Синтаксис процедуры:

```
def <имя>(<параметры>):
    <операторы>
```

Процедура начинается служебным словом **def** (define – «определить»).
Формальные параметры процедуры перечисляются через запятую.
Операторы, входящие в тело процедуры, записываются с отступом. Процедура должна быть определена до первого её вызова.
Вызов процедуры
Вызов процедуры осуществляется по её имени с указанием фактических параметров (аргументов).
<имя>(<аргументы>)

Пример:

```
def digit():                    # процедура без параметра
    print ("1111111") # основная программа выводит три строки из семи единиц

digit()  # вызовы процедуры

digit()

digit()
```

Результат:

```
1111111
1111111
1111111
```

Подпрограммы-процедуры выполняют некоторые действия, например, выводят результат на экран в определенном виде (простой пример, оператор `print()` – это стандартная подпрограмма-процедура, которая выводит данные на экран)
Надо запомнить!

1. Процедура начинается со слова def (от англ. - define - определить). После имени процедуры записаны пустые скобки и двоеточие. Внутри скобок можно указывать параметры.
2. Все операторы, которые выполняются в процедуре, записываются с отступом.
3. Чтобы выполнить процедуру, в основной программе необходимо вызвать ее **по имени и не забыть написать скобки**!
4. Вызывать процедуру в программе можно сколько угодно раз.

Использование процедур сокращает код и повышает удобочитаемость.

Функция – это вспомогательный алгоритм, который всегда возвращает в основной алгоритм значение-результат.

```
def <имя>(<параметры>):
    <операторы>
    return <результат>
```

После оператора **return** («вернуть») записывается результат, который возвращает функция. В функции может быть несколько операторов **return**, после выполнения любого из них работа функции заканчивается.

Вызов функции

Для создания функции используют ключевое слово def. Вот пример простейшей функции, которая не получает и не возвращает никаких данных – просто выполняет одну команду по выводу строки с приветствием:

```
def my_function():
    print('Привет от Python')
```

Для вызова такой функции достаточно написать ее название:

```
my_function()
```

Результат вызова:

Привет от Python

А это пример простейшей функции с параметром:

```
def my_function(name):
    print(f'Привет, {name}')
```

При вызове функция получает аргумент:

```
my_function('Вася')
```

Результат вызова:

Привет, Вася

Функции можно вызывать везде, где можно использовать выражение соответствующего типа (в операторах присваивания или вывода).

<имя>(<аргументы>)

Пример:

```
def summa(a, b) # a, b - параметры функции
    c = a+b          # вычисление функции
    return c         # возвращаемый результат
```

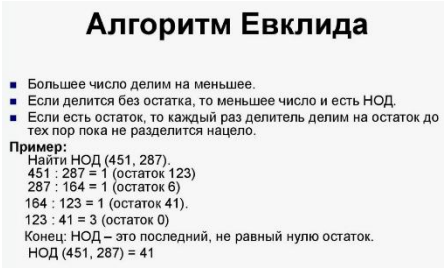
Результат:

```
s = summa(2, 3)# вызов функции
print(s)   # напечатается число 5
```

Подпрограммы-функции **возвращают результат** (число, символьную строчку и т.д.), который мы можем использовать в основной программе.

Функция выполняется только тогда, когда ее вызывает основная программа.
В функцию можно передавать различные данные. Параметры – это переменные, которые используются при объявлении функции, аргументы – фактические значения, которые передаются переменным при вызове функции.
Функции могут передавать результаты своей работы в основную программу или в другие функции.

Пример задачи:
Найти наибольший общий делитель чисел 16, 32, 40, 64, 80 и 128, используя в качестве процедуры алгоритм Евклида.



 # Процедура НОД

```
def nod (a, b):          # a, b - формальные параметры
   global x              # объявление глобальной переменной
   while a != b:         # пока числа не равны
      if a > b:          # большее заменяем разностью чисел
         a = a-b
      else:
         b = b-a
   x = a                 # результат процедуры НОД
```

 # Основная программа

```
m = [16, 32, 40, 64, 80, 128] # массив исходных чисел
print (m)                     # вывод чисел на экран
x = m[0]                      # первое число для процедуры
for i in range (1, 6):    # перебор чисел в массиве
   y = m[i]                   # второе число для процедуры
   nod (x, y)             # вызов процедуры для этих чисел
print ("НОД = ", x)            # вывод результата
```

Результат:

```
[16, 32, 40, 64, 80,
128]

НОД =  8
```

Рекурсией называется процесс, когда функция вызывает саму себя. Её можно использовать вместо циклов, например, для задачи по нахождению факториала.

```
def f(num):
   if num == 0:
      return 1
   return f(num-1) * num
print(f(5)) # Выведет число 120
```

Рекурсию рекомендуется использовать только там, где это действительно необходимо. Интерпретатор Python автоматически выделяет память для выполняющейся функции, если вызовов самой себя будет слишком много, это приведёт к переполнению стека и аварийному завершению программы. Следующий код вызовет исключение “RecursionError”, которая показывает, что превышен максимальный лимит рекурсии.

```
def x(num):
   a = num - 1
   print(a)
```

```
    x(a)
x(5)
```

Один из примеров применения рекурсии – это расчёт чисел Фибоначчи.

Пример:

```
# Функция вычисления n-го члена последовательности Фибоначчи
def f (n):
    if n==1 or n==2:          # первые два числа равны 1
        rez = 1
    else:                     # число равно сумме двух предыдущих
        rez = f(n-1) + f(n-2)  # рекурсивный вызов функции
    return rez
# Основная программа
for i in range(1, 11):    # для чисел от 1 до 10
    print (f(i), end=" ") # вывод очередного значения функции
```

Результат:

```
1 1 2 3 5 8 13 21 34 55
```

Чтобы разобраться в принципе работы рекурсивных функций, нужно понять (в самых общих чертах) концепцию **стека вызовов**.

Стек – это структура данных **LIFO** (last in, first out): информация последовательно добавляется в «стопку», каждый новый объект помещается поверх предыдущего, а извлекаются объекты в обратном порядке, – начиная с верхнего.

```
stack = []
for i in range(1, 11):
    stack.append(f'{i}-й элемент')
    print(f'+ {i}-й элемент добавлен')
    for i in stack:
        print(i, end=" ")
print('\n')
for i in range(len(stack)):
    print('В стеке: ', end=" ")
    for i in stack:
        print(i, end=" ")
    print(f'\n{stack.pop()} удален из стека')
```

Стек вызовов, в свою очередь, – это область памяти, в которой выполняются функции. При каждом вызове функции создается фрейм – фрагмент памяти, – в котором содержится:

- информация о текущем состоянии выполнения функции;
- значения всех переменных, которые функция получила для обработки;
- локальные данные, созданные во время очередного вызова;
- сведения о строке программы, к которой нужно вернуться после выполнения функции.

Фреймы помещаются в стек вызовов, как уже было показано в примере выше, и удаляются точно так же, сверху вниз. Рекурсивные функции при каждом новом вызове используют данные, созданные во время работы предыдущего вызова.

Программисту не нужно беспокоиться о работе стека вызовов – созданием фреймов и управлением стеком занимается интерпретатор. Однако понимание принципа работы стека вызовов значительно упрощает создание рекурсивных функций. Неверное же

использование рекурсии приводит к переполнению стека (**stack overflow**).

Переполнить стек в опытных целях можно с помощью простейшей рекурсивной функции, которая бесконечно вызывает сама себя, но не возвращает никаких данных и не содержит никакого условия для прекращения своей работы:

Чтобы стек вызовов не переполнялся, в каждой рекурсивной функции всегда должны быть предусмотрены два случая:

1. **Граничный**, при котором функция завершает работу и возвращает данные в основную программу.
2. **Рекурсивный**, при котором функция продолжает вызывать себя.

Вот пример простейшей рекурсивной функции, в которой учтены оба случая:

```
def greetings(st):
 print(st)
 if len(st) == 0:  # Граничный случай
     return
 else:       # Рекурсивный случай
     greetings(st[:-1])
```

Вызовы функции прекращаются, когда длина выводимой подстроки достигает **0**:

```
Hello, world!
Hello, world
Hello, worl
Hello, wor
Hello, wo
Hello, w
Hello,
Hello,
Hello
Hell
Hel
He
H
```

Основные встроенные функции в Python

Python предоставляет программисту все необходимые основные функции для работы, кроме того, используя дополнительные модули (подключая разные библиотеки), можно найти уже реализованные методы для практически любой задачи.

- **print()** Выводит объекты на экран или в файл.
  Пример использования print("output string", end="").
  Принимает аргументы:
  - object — то, что нужно вывести;
  - sep — разделитель, который ставиться между объектов, по умолчанию — пробел;

    - end — окончание после объекта, например управляющий символ “\n”;
    - file — атрибут, позволяющий передать объект в файл (по умолчанию выводит на экран);
- **len()** Возвращает количество элементов, содержащихся в кортеже, словаре или списке.
- **str()** Преобразует переданный в качестве аргумента объект в строку.
- **int()** Преобразует объект в целое число. Если передать в качестве аргумента строку, вызовется ошибка, целое число выведется без изменений, а у числа с плавающей точкой отбросится дробная часть.
- **range()** Возвращает диапазон значений, в основном используется в условии цикла for.
- **bool()** Приводит объект в логическому типу. Например, 0 — `False`, а 1 — `True`.
- **sum()** Возвращает сумму переданных элементов.
- **min() и max()** Возвращают минимальный и максимальный элемент из переданных.
- **type()** Возвращает тип объекта, обычно используется при отладке кода.
- **dir()** Возвращает список имён, доступных в локальной области видимости, или атрибуты объекта, переданного в качестве аргумента.
- **help()** Выводит информацию о переданном объекте из встроенной справочной системы. Её целесообразно использовать только в интерактивном режиме Python интерпретатора.

Практическая часть

1. Написать подпрограмму сложения, вычитания двух чисел, вывести на экран
2. Вычислить периметр и площадь прямоугольника. Значения: 4,7. Вывести на экран
3. Написать подпрограмму-процедуру, которая печатает 60 раз указанный символ (введенный с клавиатуры), каждый с новой строки. При написании использовать условный оператор.
4. Написать подпрограмму-процедуру. Найти большее из пяти заданных чисел, используя вспомогательный алгоритм нахождения большего из двух чисел.
Заданные числа: 15, 34, 46, 54, 87
5. Написать подпрограмму, в которой подсчитать количество слов в тексте, используя вспомогательный алгоритм нахождения количества пробелов в строке.
(Подсказа: Используем функцию def count_space():
Текст для ввода – «Python — это язык программирования, который широко используется в интернет-приложениях, разработке программного обеспечения, науке о данных и машинном обучении»)
6. Написать подпрограмму нахождения максимального значения чисел: 23, 54, 89
(Подсказка: использовать функцию def max():)
7. Написать подпрограмму-функцию для возведения числа n в степень m. Значения используйте любые.
8. Написать функцию, которая определяет: делится ли сумма цифр заданного числа на 3. Причем, если в полученной сумме больше одной цифры, то необходимо повторить операцию, пока сумма не будет состоять из единственной цифры.

Контрольные вопросы:

1. Для чего используются подпрограммы?
2. В чём основное различие процедур и функций?
3. Что такое рекурсивная функция?
4. Отличие глобальной переменной от локальной?
5. Что такое стек, для чего используется?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 29

Тема: Посимвольная обработка строк.

Количество часов на выполнение: 2

Цель работы: Изучить методы и приемы посимвольной обработки строк, научиться выполнять операции перебора, изменения и анализа символов, а также уметь применять эти навыки для решения практических задач в программировании.

Оборудование: Персональный компьютер

Методика выполнения задания

Теоретическая часть

Введение

Строки — это последовательности символов, используемые для хранения текста. Посимвольная обработка строк — это методы и операции, позволяющие просматривать и изменять отдельные символы строки.

Основные понятия

Индексы: позиции символов в строке, нумеруются с 0.

Длина строки: количество символов, которые она содержит.

Обработка посимвольно: выполнение операций по одному символу, например, проверка, изменение, сравнение.

Основные операции

Итерация по строке: проход по каждому символу с помощью цикла.

Доступ к символу по индексу: получить или изменить конкретный символ.

Проверка символов: например, определить, является ли символ буквой, цифрой, пробелом и т.д.

Модификация строки: создание новой строки с изменёнными символами (учитывая, что в большинстве языков строки неизменяемы).

Важные функции и методы (на примере Python)

len(s) — длина строки.

s[i] — доступ к символу по индексу.

for char in s: — перебор символов строки.

s.replace(old, new) — замена подстроки.

s.lower(), s.upper() — преобразование регистра.

Методы для проверки символов: str.isalpha(), str.isdigit(), strisspace(), и др.

Практическая часть:

Задание 1: Посимвольно вывести строку

Задача: Напишите программу, которая считывает строку и выводит каждый символ на отдельной строке.

```python
s = input("Введите строку: ")
for ch in s:
    print(ch)
```

Задание 2: Подсчёт количества гласных и согласных

Задача: Напишите программу, которая учитывает только буквы и считает гласные (а, е, ё, и, о, у, ы, э, ю, я) и согласные.

```python
s = input("Введите строку: ").lower()
vowels = "аеёиоуыэюя"
consonants = "бвгджзйклмнпрстфхцчшщ"
count_vowels = 0
count_consonants = 0

for ch in s:
    if ch.isalpha():
        if ch in vowels:
            count_vowels += 1
        elif ch in consonants:
            count_consonants += 1


print("Гласных букв:", count_vowels)
print("Согласных букв:", count_consonants)
```

Задание 3: Замена символов в строке

Задача:

```python
s = input("Введите строку: ")
result = s.replace(' ', '_')
print("Результат:", result)
```

Задание 4: Проверка на палиндром

Задача:

```python
s = input("Введите строку: ").lower()
# Удаление пробелов и знаков препинания (при необходимости)
s_clean = ''.join(ch for ch in s if ch.isalnum())
if s_clean == s_clean[::-1]:
    print("Строка является палиндромом.")
else:
    print("Строка не является палиндромом.")
```

Контрольные вопросы

1. Что такое посимвольная обработка строк?
2. Как в Python получить длину строки?

3. Как обратиться к символу строки по индексу?
4. Какие операции можно выполнить при посимвольной обработке строки?
5. Чем отличается изменение строки от её копирования?
6. Какие методы помогают определить, является ли символ буквой, цифрой или пробелом?
7. Что такое палиндром? Как проверить, является ли строка палиндромом?
8. Почему строки в большинстве языков являются неизменяемыми?
9. Какие преимущества дает посимвольная обработка строки?
10. В чем заключается основная сложность при изменении строки в языках с неизменяемыми строками?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 30

Тема: Заполнение массива.

Количество часов на выполнение: 4.

Цель работы: Познакомиться с основными понятиями Microsoft PowerPoint и приемами создания и оформления презентаций.

Оборудование: Персональный компьютер, MS Word.

Методика выполнения задания

Теоретическая часть

В Питоне нет структуры данных, полностью соответствующей массиву. Однако, есть списки, которые являются их надмножеством, то есть это те же массивы, но с расширенным функционалом. Эти структуры удобнее в использовании, но цена такого удобства, как всегда, производительность и потребляемые ресурсы. И массив, и список – это упорядоченные коллекции, но разница между ними заключается в том, что классический массив должен содержать элементы только одного типа, а список Python может содержать любые элементы.

Существует несколько способ создать массив. Ниже приведены примеры как это можно сделать. Важно не забывать: индексация массива начинается с нуля.

```
можно_так = [1, 2, 3, 4, 5]
можно_так_2 = list('итерируемый объект')
а_можно_и_так = [i for i in range(5)]
print('можно_так:', можно_так)
print('можно_так_2:', можно_так_2)
print('а_можно_и_так:', а_можно_и_так)
print('можно_так[0]:', можно_так[0])
print('а_можно_и_так[3]:', а_можно_и_так[3])
# Вывод:
можно_так: [1, 2, 3, 4, 5]
можно_так_2: ['и', 'т', 'е', 'р', 'и', 'р', 'у', 'е', 'м', 'ы', 'й', ' ', 'о', 'б', 'ъ',
а_можно_и_так: [0, 1, 2, 3, 4]
можно_так[0]: 1
а_можно_и_так[3]: 3
```

Многомерный массив

Двухмерный массив в Python можно объявить следующим образом.

```
example_array = [[-1, 0, 0, 1], [2, 3, 5, 8]]
print(example_array[0])
print(example_array[1])
print(example_array[0][3])
# Вывод:
[-1, 0, 0, 1]
[2, 3, 5, 8]
1
```

Точно так же мы можем определить трехмерный массив или многомерный массив в Питоне.

```
example_array = [[[-1, 0], [0, 1]], [[2, 3], [5, 8]]]
print(example_array[0])
print(example_array[1])
print(example_array[0][1])
print(example_array[0][1][0])
# Вывод:
[[-1, 0], [0, 1]]
[[2, 3], [5, 8]]
[0, 1]
0
```

Операции с массивами

Мы можем использовать цикл for для обхода элементов массива.

```
example_array = [1, 2, 3]
for i in range(len(example_array)):
```

```
    print(example_array[i])
# Вывод:
1
2
3
```

Здесь представлен подход, свойственный большинству языков программирования. В Python же есть свой, более удобный вариант.

```
example_array = [1, 2, 3]
for i in example_array:
    print(i)
# Вывод:
1
2
3
```

Обход многомерного массива

Для того чтоб получить элементы многомерного массива придётся использовать вложенные циклы.

```
example_array = [[1, 2], [3, 4]]
for i in example_array:
    for x in i:
        print(x)
# Вывод:
1
2
3
4
```

Добавление

Мы можем использовать функцию insert() для вставки элемента по указанному индексу. Элементы из указанного индекса сдвигаются вправо на одну позицию.

```
example_array = [[1, 2], [3, 4]]
example_array.insert(0, -1)
example_array.insert(2, [-1, 13, 64])
print(example_array)
# Вывод:
[-1, [1, 2], [-1, 13, 64], [3, 4]]
```

Если вам нужно добавить элемент в конец массива, используйте функцию append().

```
example_array = [[1, 2], [3, 4]]
example_array.append(-1)
example_array.append([-1, 13, 64])
print(example_array)
# Вывод:
[[1, 2], [3, 4], -1, [-1, 13, 64]]
```

Для того, чтоб объединить элементы двух массивов используйте метод extend.

```
example_array = [1, 2, 3, 4]
example_array.extend([5, 6])
print(example_array)
# Вывод:
[1, 2, 3, 4, 5, 6]
```

Определение размера

Используйте метод len() чтобы вернуть длину массива (число элементов массива). Не стоит путать размер массива с его размерностью!

```
example_array = [[1, 2], [3, 4]]
print('Размер массива:', len(example_array))
example_array.append(-1)
print('Размер массива:', len(example_array))
example_array.append([-1, 13, 64])
print('Размер массива:', len(example_array))
# Вывод:
Размер массива: 2
Размер массива: 3
Размер массива: 4
```

Поскольку индексация элементов начинается с нуля, длина массива всегда на единицу больше, чем индекс последнего элемента.

```
example_array = [[1, 2], [3, 4]]
print('Равна ли длина массива номеру последнего элемента + 1?',
len(example_array) is (example_array.index(example_array[-1]) + 1))
example_array.append(-1)
print('Увеличили размер массива.')
print('Равна ли теперь длина массива номеру последнего элемента + 1?',
len(example_array) is (example_array.index(example_array[-1]) + 1))
# Вывод:
Равна ли длина массива номеру последнего элемента + 1? True
Увеличили размер массива.
Равна ли теперь длина массива номеру последнего элемента + 1? True
```

Небольшое пояснение: метод списка .index() возвращает индекс элемента, значение которого совпадает с тем, которое передали методу. Здесь мы передаём значение последнего элемента и, таким образом, получаем индекс последнего элемента. Будьте осторожны: если в списке есть повторяющиеся значения, этот приём не сработает!

Срез

Срез Python предоставляет особый способ создания массива из другого массива.

```
example_array = [[1, 2], [3, 4]]
print(example_array[::-1])
print(example_array[1:])
```

```
print(example_array[0][:-1])
# Вывод:
[[3, 4], [1, 2]]
[[3, 4]]
[1]
```

Функция pop

В Python удалить ненужные элементы из массива можно при помощи метода pop, аргументом которого является индекс ячейки. Как и в случае с добавлением нового элемента, метод необходимо вызвать через ранее созданный объект.

```
example_array = [1, 2, 6, 3, 4]
print(example_array.pop(4))
print(example_array)
# Вывод:
4
[1, 2, 6, 3]
```

После выполнения данной операции содержимое массива сдвигается так, чтобы количество доступных ячеек памяти совпадало с текущим количеством элементов.

Методы массива

В Python есть набор встроенных методов, которые вы можете использовать при работе с list.

Метод	**Значение**
append()	Добавляет элементы в конец списка
clear()	Удаляет все элементы в списке
copy()	Возвращает копию списка
count()	Возвращает число элементов с определенным значением
extend()	Добавляет элементы списка в конец текущего списка
index()	Возвращает индекс первого элемента с определенным значением
insert()	Добавляет элемент в определенную позицию
pop()	Удаляет элемент по индексу
remove()	Убирает элементы по значению
reverse()	Разворачивает порядок в списке

sort()	Сортирует список

Модуль array

Если Вам всё-таки нужен именно классический массив, вы можете использовать встроенный модуль array. Он почти не отличается от структуры list, за исключением, пожалуй, объявления.

Вот небольшая демонстрация:

```
import array

example_array = array.array('i', [1, 2, 6, 3, 4]) # превый аргумент указывает на тип элементов. i означает integer
example_array.insert(0, -1)
print('После вставки:', example_array)
example_array.append(-1)
print('После добавления в конец:', example_array)
example_array.extend([5, 6])
print('После объединения со списком:', example_array)
print('Удалён элемент:', example_array.pop(4))
print('После удаления элемента:', example_array)
print('Срез:', example_array[0:4])
# Вывод:
После вставки: array('i', [-1, 1, 2, 6, 3, 4])
После добавления в конец: array('i', [-1, 1, 2, 6, 3, 4, -1])
После объединения со списком: array('i', [-1, 1, 2, 6, 3, 4, -1, 5, 6])
Удалён элемент: 3
После удаления элемента: array('i', [-1, 1, 2, 6, 4, -1, 5, 6])
Срез: array('i', [-1, 1, 2, 6])
```

Типы элементов массива

Элементы массива в модуле array могут быть следующих типов:

Код типа	**Тип в C**	**Тип в python**
‘b’	signed char	int
‘B’	unsigned char	int
‘h’	signed short	int
‘H’	unsigned short	int
‘i’	signed int	int
‘I’	unsigned int	int
‘l’	signed long	int
‘L’	unsigned long	int
‘q’	signed long long	int
‘Q’	unsigned long long	int
‘f’	float	float
‘d’	double	float

Как Вы можете видеть, со строками модуль не работает.

Практическая часть

Задание 1. Создайте массив размером 40 и заполните его через цикл.

Задание 2. Создайте массив размером 20 и заполните его случайными значениями в диапазоне от -2 до 8.

Задание 3. Создайте массив размером 160, заполните его случайными значениями в диапазоне от 2 до 8. Выведите на экран получившейся массив, после чего удалите все элементы массива равные значению 2 и 5. Сравните размер массива.

Задание 4. Дан массив чисел. Превратите его в массив квадратов этих чисел.

Задание 5. Дан массив некоторых целых чисел, найдите значение 30 в нем и, если оно присутствует, замените его на 300. Обновите список только при первом вхождении числа 30.

Контрольные вопросы:

1. Что такое массив в python?
2. Как вывести массив на экран?
3. Как заполнит массив случайными числами?
4. Что такое многомерный массив?
5. С помощью чего создаётся многомерный массив?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №31

Тема: Простые методы сортировки массива.

Количество часов на выполнение: 4

Цель работы: познакомиться с массивами в Python, с методами сортировки

Оборудование: Персональный компьютер , Python

Методика выполнения задания

Теоретическая часть

Сортировка означает размещение элементов в определенном порядке. Этот конкретный порядок определяется свойством сравнения элементов. В случае целых чисел мы говорим, что сначала идет меньшее число, а потом — большее.

Расположение элементов в определенном порядке улучшает поиск элемента. Следовательно, сортировка широко используется в информатике.

В данной статье мы рассмотрим обычные алгоритмы сортировки и их реализации на Python. Для сравнения их производительности мы будем рассматривать задачу с сайта Leetcode о сортировке массива. Размеры данных этой задачи ограничены следующим образом:

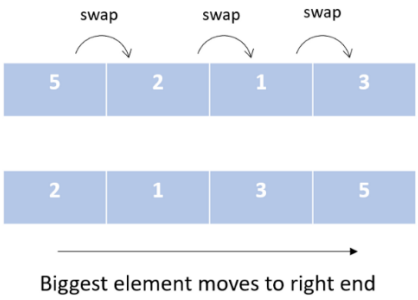
```
1 <= nums.length <= 50000
-50000 <= nums[i] <= 50000
```

Мы решили эту задачу при помощи всех известных алгоритмов сортировки. Вот какие у нас получились результаты:

Sl. No.	Algorithm	Runtime (ms)	Memory (MB)
1	Сортировка пузырьком	TLE	NA
2	Сортировка выбором	TLE	NA
3	Сортировка вставками	TLE	NA
4	Сортировка Шелла	408	20.2
5	Пирамидальная сортировка	532	20.1
6	Сортировка слиянием	388	21.8
7	Быстрая сортировка	236	21.5
8	Сортировка подсчетом	140	23.2

Сортировка методом пузырька

Это самый простой алгоритм сортировки. В процессе его выполнения мы перебираем наш список и на каждой итерации сравниваем элементы попарно. При необходимости элементы меняются местами, чтобы больший элемент отправлялся в конец списка.



Алгоритм сортировки пузырьком:

нерекурсивный;

устойчивый;

преобразует входные данные без использования вспомогательной структуры данных (in place);

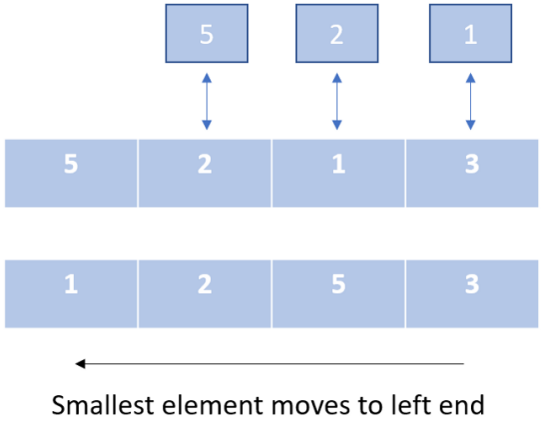
имеет сложность $O(n^2)$;

```
def bubbleSort(array):
    swapped = False
    for i in range(len(array)-1,0,-1):
        for j in range(i):
            if array[j]>array[j+1]:
                array[j], array[j+1] = array[j+1], array[j]
                swapped= True
        if swapped:
            swapped=False
        else:
            break
    return array
```

Сортировка выбором

В этом алгоритме мы создаем два сегмента нашего списка: один отсортированный, а другой несортированный.

В процессе выполнения алгоритма мы каждый раз удаляем самый маленький элемент из несортированного сегмента списка и добавляем его в отсортированный сегмент. Мы не меняем местами промежуточные элементы. Следовательно, этот алгоритм сортирует массив с минимальным количеством перестановок.



Алгоритм сортировки выбором:

нерекурсивный;

может быть как устойчивым, так и неустойчивым;

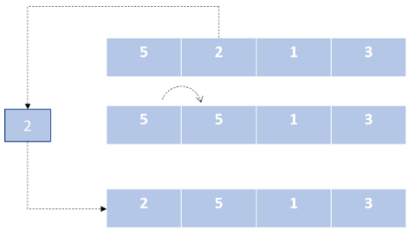
преобразует входные данные без использования вспомогательной структуры данных (in place);

имеет сложность $O(n^2)$;

```
def selectionSort(array):
    for i in range(len(array)-1):
        min_idx = i
        for idx in range(i + 1, len(array)-1):
            if array[idx] < array[min_idx]:
                min_idx = idx
        array[i], array[min_idx] = array[min_idx], array[i]
    return array
```

Сортировка вставками

Подобно алгоритму сортировки выбором, мы делим наш список на две части. Далее мы перебираем неотсортированную часть и вставляем каждый элемент из данного сегмента на его правильное место в отсортированной части списка.



Алгоритм сортировки вставками:

нерекурсивный;

устойчивый;

преобразует входные данные без использования вспомогательной структуры данных (in place);

имеет сложность O(n2);

```
def insertionSort(array):
    for i in range(1, len(array)):
        key = array[i]
        j = i-1
        while array[j] > key and j >= 0:
            array[j+1] = array[j]
            j -= 1
        array[j+1] = key
    return array
```

Пирамидальная сортировка

Также известна как сортировка кучей. Этот популярный алгоритм, как и сортировки вставками или выборкой, сегментирует список на две части: отсортированную и неотсортированную. Алгоритм преобразует второй сегмент списка в структуру данных «куча» (heap), чтобы можно было эффективно определить самый большой элемент.

Алгоритм

Сначала преобразуем список в Max Heap — бинарное дерево, где самый большой элемент является вершиной дерева. Затем помещаем этот элемент в конец списка. После перестраиваем Max Heap и снова помещаем новый наибольший элемент уже перед последним элементом в списке.

Этот процесс построения кучи повторяется, пока все вершины дерева не будут удалены.

Реализация

Создадим вспомогательную функцию heapify() для реализации этого алгоритма:

```
def heapify(nums, heap_size, root_index):
    # Индекс наибольшего элемента считаем корневым индексом
    largest = root_index
    left_child = (2 * root_index) + 1
    right_child = (2 * root_index) + 2

    # Если левый потомок корня — допустимый индекс, а элемент больше,
    # чем текущий наибольший, обновляем наибольший элемент
    if left_child < heap_size and nums[left_child] > nums[largest]:
        largest = left_child

    # То же самое для правого потомка корня
    if right_child < heap_size and nums[right_child] > nums[largest]:
        largest = right_child

    # Если наибольший элемент больше не корневой, они меняются местами
    if largest != root_index:
        nums[root_index], nums[largest] = nums[largest], nums[root_index]
        # Heapify the new root element to ensure it's the largest
        heapify(nums, heap_size, largest)
```

```
def heap_sort(nums):
    n = len(nums)

    # Создаём Max Heap из списка
    # Второй аргумент означает остановку алгоритма перед элементом -1, т.е.
    # перед первым элементом списка
    # 3-й аргумент означает повторный проход по списку в обратном направлении,
    # уменьшая счётчик i на 1
    for i in range(n, -1, -1):
        heapify(nums, n, i)

    # Перемещаем корень Max Heap в конец списка
    for i in range(n - 1, 0, -1):
        nums[i], nums[0] = nums[0], nums[i]
        heapify(nums, i, 0)

# Проверяем, что оно работает
random_list_of_nums = [35, 12, 43, 8, 51]
heap_sort(random_list_of_nums)
print(random_list_of_nums)
```

Время сортировки

В среднем время сортировки кучей составляет O(n log n), что уже значительно быстрее предыдущих алгоритмов.

Сортировка слиянием

Этот алгоритм относится к алгоритмам «разделяй и властвуй». Он разбивает список на две части, каждую из них он разбивает ещё на две и т. д. Список разбивается пополам, пока не останутся единичные элементы.

Соседние элементы становятся отсортированными парами. Затем эти пары объединяются и сортируются с другими парами. Этот процесс продолжается до тех пор, пока не отсортируются все элементы.

Алгоритм

Список рекурсивно разделяется пополам, пока в итоге не получатся списки размером в один элемент. Массив из одного элемента считается упорядоченным. Соседние элементы сравниваются и соединяются вместе. Это происходит до тех пор, пока не получится полный отсортированный список.

Сортировка осуществляется путём сравнения наименьших элементов каждого подмассива. Первые элементы каждого подмассива сравниваются первыми. Наименьший элемент перемещается в результирующий массив. Счётчики результирующего массива и подмассива, откуда был взят элемент, увеличиваются на 1.

Реализация

```
def merge(left_list, right_list):
    sorted_list = []
    left_list_index = right_list_index = 0

    # Длина списков часто используется, поэтому создадим переменные для удобства
    left_list_length, right_list_length = len(left_list), len(right_list)

    for _ in range(left_list_length + right_list_length):
        if left_list_index < left_list_length and right_list_index < right_list_length:
            # Сравниваем первые элементы в начале каждого списка
            # Если первый элемент левого подсписка меньше, добавляем его
            # в отсортированный массив
            if left_list[left_list_index] <= right_list[right_list_index]:
                sorted_list.append(left_list[left_list_index])
                left_list_index += 1
            # Если первый элемент правого подсписка меньше, добавляем его
            # в отсортированный массив
            else:
                sorted_list.append(right_list[right_list_index])
                right_list_index += 1
```

```
        # Если достигнут конец левого списка, элементы правого списка
        # добавляем в конец результирующего списка
        elif left_list_index == left_list_length:
            sorted_list.append(right_list[right_list_index])
            right_list_index += 1
        # Если достигнут конец правого списка, элементы левого списка
        # добавляем в отсортированный массив
        elif right_list_index == right_list_length:
            sorted_list.append(left_list[left_list_index])
            left_list_index += 1

    return sorted_list

def merge_sort(nums):
    # Возвращаем список, если он состоит из одного элемента
    if len(nums) <= 1:
        return nums
```

```
    # Для того чтобы найти середину списка, используем деление без остатка
    # Индексы должны быть integer
    mid = len(nums) // 2

    # Сортируем и объединяем подсписки
    left_list = merge_sort(nums[:mid])
    right_list = merge_sort(nums[mid:])

    # Объединяем отсортированные списки в результирующий
    return merge(left_list, right_list)

# Проверяем, что оно работает
random_list_of_nums = [120, 45, 68, 250, 176]
random_list_of_nums = merge_sort(random_list_of_nums)
print(random_list_of_nums)
```

Обратите внимание, что функция merge_sort(), в отличие от предыдущих алгоритмов, возвращает новый список, а не сортирует существующий. Поэтому такая сортировка требует больше памяти для создания нового списка того же размера, что и входной список.

Время сортировки

В среднем время сортировки слиянием составляет O(n log n).

Практическая часть

Задание 1. Предположим, что вам нужно отсортировать, Ваш плейлист по количеству прослушиваю композиции, т.е. если песню №15 Вы слушали чаще всего, то она должна оказаться вверху списка

Алгоритм сортировки выбором:

1. Мы смотрим, сколько раз мы слушали каждую песню
2. Ищем песню с максимальным количеством прослушиваний
3. Кидаем ее вверх списка
4. Начинаем чтение списка уже с 2 элемента, потому что на первом месте уже есть нужная нам песня
5. Все эти действия выполняем, пока список не будет весь отсортирован

Ура, ваш плейлист отсортирован!

Данный алгоритм можно применять для сортировки фамилий в телефонной книге, сортировать даты различных событий и даже отсортировать email сообщения по дате

```
128 # Сортировка выбором
129 def findSmallest(arr: list) -> int:  # Функция для нахождения минимума
130     smallest = arr[0]  # Допустим первый элемент списка - это минимум
131     smallest_index = 0
132     for i in range(1, len(arr)):  # Пробегаемся по всему списку
133         if arr[i] < smallest:  # если видим, что есть элемент меньше, то это теперь нвоый минимум
134             smallest = arr[i]
135             smallest_index = i
136     return smallest_index
137
138
139 # noinspection SpellCheckingInspection
140 def selectionSort(arr: list) -> list:  # Функция для сортировки списка по не убывыанию
141     newArr = []
142     for i in range(len(arr)):
143         smallest = findSmallest(arr)  # Мы для каждой иттерации находим минимум
144         newArr.append(arr.pop(smallest))  # Затем добавляем в новый
145                                           # список минимум и удаляем его из старого
146     return newArr
147 # Бинго, мы отсортировали список!
```

Задание 2. Дан список из N ($N \leq 2*105$) элементов,которые принимают целые значения от 0 до 100. Отсортируйте этот список в порядке неубывания элементов. Выведите полученный список. Решение оформите в виде функции CountSort(A), которая модифицирует передаваемый ей список.

Примечание

Сложность работы программы должна быть $O(n)$. Использование встроенной сортировки(sort, sorted), алгоритмов сортировки пузырёк/quick sort/merge sort и других запрещено!

Примеры
Входные данные
7 3 4 2 5
Выходные данные
2 3 4 5 7

Задание 3. Считайте со стандартного ввода символ и выведите его код.

Входные данные
Программа получает на вход один символ с кодом от 33 до 126.

Выходные данные
Нужно вывести одно число - код считанного символа

Примеры
Входные данные
A
Выходные данные
65

Контрольные вопросы:

1. Какой алгоритм сортировки считается самым простым?
2. Какой метод используется для сортировки массива?
3. Какой метод сортировки используется в Python?
4. Какие бывают сортировки массива?
5. Какой самый быстрый метод сортировки?
6. Какой метод сортировки самый распространенный?
7. Что такое сортировка в Питоне?
8. Какая команда используется для сортировки данных?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:
Отчет должен содержать:
1.Название работы.
2.Цель работы.
3.Результаты выполнения заданий.
4. Ответы на контрольные вопросы.
Практическая работа сохраняется в личной папке на сетевом диске.
Затем прикрепляется в Электронный курс https://el.istu.edu Информатика
Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №32

Тема: Применение списков и словарей в реальных задачах.

Количество часов на выполнение: 4

Цель работы: получение практических навыков программирования в задании словарей и выполнении операций над ними.

Оборудование: Персональный компьютер, ПК, PyCharm или Visual Studio Code

Методика выполнения задания
Теоретическая часть

Словарь в Python (dict) - неупорядоченная последовательность произвольного типа с доступом по ключу. Их иногда ещё называют ассоциативными массивами или хеш-таблицами.

Элементами такой коллекции выступают пары объектов, каждая из которых включает в себя **ключ** и **значение**. Это позволяет идентифицировать ее элементы не по числовому индексу (как списки), а по произвольному, т.е. в качестве идентификатора можно использовать и число, и текстовую строку. Значениями в словаре, в отличие от ключей, могут быть объекты любого типа – числа, строки, кортежи, списки и даже другие словари.

В Python словари пишутся в фигурных скобках **{}**.

Создание словаря:

1) присвоить свободной переменной произвольное количество пар объектов. Элементы необходимо поместить в фигурные скобки, а между ключом и значением должен стоять символ двоеточия**.**

```
d = {} # создание пустого словаря
thisdict = {
"brand": "Ford",
"model": "Mustang","year":
1964
}
print(thisdict)          # {'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

2) использовать **конструктор dict().** В таком случае быть ключом может только строка, как это показано в следующем примере кода.

```
d = dict()                          # создание пустого словаря
thisdict = dict(brand="Ford", model="Mustang", year=1964)
print(thisdict) # {'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

Обратите внимание, ключевые слова не являются строками, а вместо двоеточия для задания используется «равно».

Для того чтобы ввести в коллекцию новую пару объектов необходимо указать новый ключ в квадратных скобках, а также соответствующее ему значение**.**

```
thisdict = {
"brand": "Ford",
"model": "Mustang","year":
1964
}
thisdict["color"] = "red"
print(thisdict)       # {'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}
```

Как и с другими наборами данных, производить заполнение словарей можно

при помощи генераторов**.**

```
a = {a: a * a for a in range(5)}
print(a)                    #   {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```

Для работы со словарями доступны функции, меняющие их содержимое и выполняющие различные операции над ними. Его можно конвертировать в другие типы данных, например, в строку (потребуется функция str).

```
a = {1: "one", 2: "two", 3: "three"}b
= str(a)
print(b)                    #   {1: 'one', 2: 'two', 3: 'three'}
```

В Python словари могут быть ***вложенными***, то есть выступать частью другого более крупного словаря**.**

```
a = {
"First": {
1: "one",
2: "two",
3: "three"
},
"Second": {
4: "four",
5: "five"
}
}
print(a)  # {'First': {1: 'one', 2: 'two', 3: 'three'}, 'Second': {4: 'four', 5: 'five'}}
```

В примере, создается словарь **a**, включающий в себя два других словаря (First и Second), т.е. в свою очередь, содержат несколько пар ключей и значений.

В Python существует набор встроенных методов, с помощью которых можно работать со словарями (таблица 1).

Таблица 1 – Методы словаря

Метод	Значение
len()	Получение размера (возвращает число пар)
clear()	Удаляет все элементы из словаря
copy()	Делает копию словаря
fromkeys()	Возвращает словарь с указанными ключами и значениями
get()	Возвращает значение по ключу
items()	Возвращает список, содержащий tuple для каждой пары ключ-значение
Метод	Значение
keys()	Возвращает список, содержащий ключи словаря

pop()	Удаляет элементы по ключу и возвращает его
popitem()	Удаляет последнюю пару ключа со значением
setdefault()	Задает значение по ключу. Если ключа нет в словаре, добавляет егос указанным значением или None
update()	Обновляет словарь, добавляя пары ключ-значение (объединение словарей)
values()	Возвращает список всех значений в словаре
del()	Удаляет пару по ключу или весь словарь
in	Проверяет наличие значения по ключу
eval()	Конвертация строки в словарь

На примере программы — телефонной книги покажем некоторые операции со словарями:

```
telbook = {'sasha': "32-11-4", 'vanya': '44-65-99'} # Объявляем словарь
telbook['fedya'] = '22-47-32' # добавляем новый объект в словарь print
telbook # Выводим все значения словаря
{'vanya': '44-65-99', 'fedya': '22-47-32', 'sasha': '32-11-4'}
print(len(telbook))          # определяет размер
словаря3
print telbook['vanya'] # Выводим номер значения
'vanya'44-65-99
del telbook['sasha'] # удаляем значение
'sasha'print telbook # смотрим, что
получилось
{'vanya': '44-65-99', 'fedya': '22-47-32'}
print telbook.keys() # Выводим значения по
ключу['vanya', 'fedya']
print telbook.has_key('vanya') # проверяем, есть ли в словаре значение 'vanya'
True
```

Практическая часть

Задание 1. Напишите программу, которая поможет находить номера телефонов по имени.

В первой строке задано одно целое число - количество номеров телефонов. В следующих строках заданы телефоны и имена их владельцев через пробел. В следующей строке записан запрос — это имя, чей телефон нужно найти.

Вывести номер телефона согласно запросу. Если в телефонной книге нет телефонов человека с таким именем, выведите в соответствующей строке «Нет в телефонной книге».

Задание 2. Составить программу, согласно полученному варианту задания. Ввод данных сопровождать соответствующими запросами, а вывод - наименованиями выводимых переменных.

Вариант	Задание
1	Составьте словарь о студентах. В первой строчке записано целое число – количество студентов. В следующих N строчках записана информация о студентах. Каждая строчка состоит из трёх частей, разделённых пробелом – фамилии студента, названия специальности и номере группы. В следующей строке записан запрос — это название специальности. Выведите через запятую фамилии всех студентов, обучающихся на этой специальности. Если таких фамилий нет, выведите фразу «Проверьте запрос».
2	Составьте словарь специальностей. В первой строчке записано целое число – количество специальностей. В следующих N строчках записана информация о специальностях. Каждая строчка состоит из названия специальности и разделённых тире номеров групп обучающихся этой специальности, перечисленных через запятую. В следующей строке записан запрос — это номер группы. Для каждого из запроса выведите название специальности, к которой относится группа. Если номера группы нет в словаре, оставьте строку ответа пустой.
3	Составьте толковый словарь. В первой строке задаётся целое число — количество записей в словаре. В каждой из следующих N строк дано по одной записи: сначала идёт слово, а затем через тире описание его значения. Все слова, значение которых записано в словаре, различны. В следующей строке записан запрос — это слово, значение которого нужно найти. Выведите описание его значения из словаря или фразу «Нет в словаре», если такого слова нет в словаре.
4	Составьте словарь «График отпусков» для специалиста отдела кадров. По известному графику отпусков научитесь определять, у кого отпуск в заданном месяце. В первой строчке записано целое число – количество сотрудников. В следующих N строчках записана информация о дате их отпуска. Каждая строчка состоит из трёх частей, разделённых пробелом – фамилии сотрудника, дня и месяца его отпуска. В следующей строке записан запрос — это название месяца. Выведите через пробел фамилии всех сотрудников, у которых отпуск в указанном месяце. Если в заданном месяце никто не идет в отпуск, оставьте строку ответа пустой.
5	Составьте географический словарь. В первой строчке записано целое число – количество стран. В следующих N строчках записана информация о странах. Каждая строчка состоит из названия страны и разделённых пробелом названий городов этой страны, перечисленных через запятую. В следующей строке записан запрос — это название города. Для каждого из запроса выведите название страны, в котором находится данный город. Если названия города нет в словаре, оставьте строку ответа пустой

6	Задана строка, состоящая из одного слова. Создайте словарь из строки следующим образом: в качестве ключей возьмите буквы строки, а значениями пусть будут числа, соответствующие количеству вхождений данной буквы в строку.
7	По известному списку дней рождения научитесь определять, у кого день рождения в заданном месяце. В первой строчке записано целое число — количество одноклассников. В следующих N строчках записана информация об их днях рождения. Каждая строчка состоит из трёх частей, разделённых пробелом — имени одноклассника, дня и месяца его рождения. В следующей строке записан запрос — это название месяца. Выведите через пробел имена всех одноклассников, которые родились в указанном месяце. Если в заданном месяце никто не родился, оставьте строку ответа пустой.
8	Даны два списка одинаковой длины. Необходимо создать из них словарь таким образом, чтобы элементы первого списка были ключами, а элементы второго — соответственно значениями нашего словаря.
9	По известному списку дней рождения научитесь определять, у кого день рождения в заданном месяце и какого числа. В первой строчке записано целое число — количество одноклассников. В следующих N строчках записана информация об их днях рождения. Каждая строчка состоит из трёх частей, разделённых пробелом — имени одноклассника, дня и месяца его рождения. В следующей строке записан запрос — это название месяца. Выведите через пробел имена всех одноклассников и дни их рождения, которые родились в указанном месяце. Если в заданном месяце никто не родился, оставьте строку ответа пустой.
10	Даны два списка одинаковой длины. Необходимо создать из них словарь таким образом, чтобы элементы второго списка были ключами, а элементы первого — соответственно значениями нашего словаря.

Задание 3. Дан русский текст. Текст может состоять из любых символов, вам необходимо транслитерировать (то есть заменить все русские буквы на английские) только русские буквы, а остальные оставить на месте. Строчные буквы заменяются на строчные, заглавные заменяются на заглавные. Если заглавная буква превращается при транслитерации в несколько букв, то заглавной должна остаться только первая из них (например, «Ц» → «Tc»).

Правила трансляции:

А – A
Б - B
В - V
Г - G
Д - D
Е - E
Ё - E
Ж -ZH
З - Z
И - I
Й - I
К - K

Л - L
М - M
Н - N
О - O
П - P
Р - R
С - S
Т - T
У - U
Ф - F
Х - KH
Ц – TC
Ч – CH
Ш – SH
Щ – SHCH
Ы - Y
Ъ -
Ь -
Э -E
Ю - IU
Я - IA

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 33

Тема: Поиск простых чисел в заданном диапазоне.

Количество часов на выполнение: 2.

Цель работы: решение задач длинной арифметики, реализация вычислений с многоразрядными числами.

Оборудование: Персональный компьютер , Python, любой редактор разработки (Sublime text, VS code, Notepad, Pycharm).

Методика выполнения задания
Теоретическая часть

Многоразрядные числа состоят **из трех и более цифр**. Например, число 1234 является многоразрядным числом с разрядностью 4. В таких числах каждая цифра занимает свою позицию и имеет свое значение, а разрядность является суммой разрядностей каждой цифры. Понимание разрядности числа помогает выполнять различные операции с

числами, а также упрощает процесс сложения, вычитания, умножения или деления чисел.

Многоразрядные целые числа, или числа большой длины, используются для представления и работы с очень большими числами, которые не могут быть представлены использованием обычных типов данных, таких как int или long. В общем случае, числа большой длины могут иметь произвольное количество разрядов и могут быть положительными или отрицательными.

Данные числа представляются в виде массива или списка цифр, где каждый элемент массива или элемент списка представляет один разряд числа. Например, число 123456789 будет представлено в виде массива [1, 2, 3, 4, 5, 6, 7, 8, 9]. Каждая цифра представлена отдельным элементом, что позволяет представлять числа любой длины.

Длинная арифметика

Длинная арифметика обеспечивает возможность выполнения математических операций с числами большой длины, например, сложение, вычитание, умножение, деление, возведение в степень и извлечение корня. Операции выполняются по разрядам числа, начиная с самого младшего разряда и двигаясь в старшие разряды. При выполнении операций, для учета переносов и заемов используются алгоритмы, полагающиеся на основные математические свойства чисел.

Задачи длинной арифметики широко используются в различных областях, включая криптографию, вычислительную математику, анализ данных, обработку изображений и большие вычисления. Они позволяют работать с очень большими числами, которые требуют точности и точных результатов, а также обеспечивают возможность работы с числами, которые не могут быть представлены в ограниченных типах данных.

Важно отметить, что задачи длинной арифметики могут быть вычислительно сложными из-за большого количества операций, которые необходимо выполнить для каждой операции. Поэтому эффективные алгоритмы и структуры данных используются для оптимизации производительности операций с числами большой длины.

Практическая часть

Примеры задач реализации вычислений с многоразрядными числами

Задача 1. Напишите код программы на Python. Реализуйте функцию для сложения двух многоразрядных чисел, представленных в виде строк.

Решение: (набрать код в редакторе)

```python
def add_multidigit_numbers(num1, num2):
    carry = 0
    result = []

    # Выравнивание длины чисел, добавление нулей в начало
    if len(num1) > len(num2):
        num2 = num2.zfill(len(num1))
    else:
        num1 = num1.zfill(len(num2))

    # Сложение цифр справа налево
    for i in range(len(num1) - 1, -1, -1):
        digit_sum = int(num1[i]) + int(num2[i]) + carry
        digit = digit_sum % 10
        carry = digit_sum // 10
        result.insert(0, str(digit))

    # Добавление остатка
    if carry:
        result.insert(0, str(carry))

    return "".join(result)
num1 = "12345"
num2 = "9876"
sum = add_multidigit_numbers(num1, num2)
print(sum)
```

Объяснить, почему получился такой ответ

Задача 2. Вывести все простые числа в диапазоне Python
Решение. Простое число — это натуральное число, которое больше 1 и не имеет положительного делителя, кроме 1 и самого себя, например 2, 3, 5, 7, 11, 13 и так далее.

Пользователю даются два целых числа, нижнее значение и верхнее значение. Задача состоит в том, чтобы написать программу Python для вывода всех простых чисел в заданном интервале (или диапазоне).

Чтобы напечатать все простые числа в заданном интервале, пользователь должен выполнить следующие шаги:

Шаг 1: Перебрать все элементы в заданном диапазоне.
Шаг 2: Проверить для каждого числа, есть ли у него какой-либо множитель между 1 и самим собой.
Шаг 3: Если да, то число не простое, и оно перейдет к следующему числу.
Шаг 4: Если нет, то это простое число, и программа распечатает его и проверит следующее число.
Шаг 5: Цикл прервется, когда будет достигнуто верхнее значение.
Пример: код Python для печати простого числа в заданном интервале.
Наберите этот код

```python
lower_value = int(input("введите наименьшее значение диапазона: "))
upper_value = int(input("введите верхнее значение диапазона: "))

print("Простыми числами в этом диапазоне являются: ")
for number in range(lower_value, upper_value + 1):
    if number > 1:
        for i in range(2, number):
            if(number % i) == 0:
                break
        else:
            print(number)
```

После написания программы, посмотрите результат, вставьте в отчет
Задача 3. Напишите код программы. Есть две строки, представляющие два многоразрядных числа. Затем функция выполняет вычитание чисел, используя столбиковый метод, и возвращает результат в виде строки.
Решение:

```python
def subtract_strings(num1, num2):
    # Проверяем, какое число больше
    if len(num1) < len(num2):
        num1, num2 = num2, num1

    # Добавляем нули в начале строки, чтобы оба числа имели одинаковую длину
    num2 = num2.zfill(len(num1))

    # Преобразуем строки в списки цифр
    digits1 = [int(d) for d in num1]
    digits2 = [int(d) for d in num2]

    # Выполняем вычитание
    result = []
    carry = 0
    for i in range(len(digits1)-1, -1, -1):
        diff = digits1[i] - digits2[i] - carry

        if diff < 0:
            diff += 10
            carry = 1
        else:
            carry = 0

        result.insert(0, str(diff))

    # Удаляем ведущие нули
    while result[0] == '0' and len(result) > 1:
        result.pop(0)

    # Возвращаем результат в виде строки
    return ''.join(result)


# Пример использования
num1 = input("Введите первое число: ")
num2 = input("Введите второе число: ")

result = subtract_strings(num1, num2)
print("Результат вычитания:", result)
```

После написания программы, посмотрите результат, вставьте в отчет
Задача 4. Напишите программу поиска простых чисел в диапазоне от 1 до 100:
Код определяет функцию `is_prime(n)`, которая проверяет, является ли число `n` простым. Затем создается пустой список `primes`, и для каждого числа в диапазоне от 1 до 100 производится проверка на простоту с использованием функции `is_prime()`. Если число является простым, оно добавляется в список `primes`.

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

primes = []
for num in range(1, 101):
    if is_prime(num):
        primes.append(num)

print(primes)
```

После написания программы, посмотрите результат, вставьте в отчет

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 34

Тема: Использование деревьев для вычисления арифметических выражений.

Количество часов на выполнение: 2.

Цель работы: уметь осуществлять построение дерева для заданного арифметического выражения.

Оборудование: Персональный компьютер, Python, любой редактор разработки (Sublime text, VS code, Notepad, Pycharm).

Методика выполнения задания

Теоретическая часть

Краткие теоретические сведения.

Двоичное дерево

jupiter notebook

1) пустая структура — это двоичное дерево;

2) двоичное дерево — это корень и два связанных с ним отдельных двоичных дерева (левое и правое поддеревья).

Применение:

- поиск в большом массиве данных;
- сортировка данных;

- вычисление арифметических выражений;
- оптимальное кодирование данных (метод сжатия Хаффмана)

Опишем дерево из примера с помощью словаря Python:

In [60]:

```
tree = {
    'val': 6,
    'right':{
        'val': 8,
        'right': { 'val': 9, 'right': {}, 'left': {} },
        'left': { 'val': 7, 'right': {}, 'left': {} } },
    'left':{
        'val': 3,
        'right': { 'val': 4, 'right': {}, 'left': {} },
        'left': { 'val': 1, 'right': {}, 'left': {} } } }
```

Обход и вывод дерева

Обойти дерево — это значит «посетить» все узлы по одному разу. Существуют несколько способов обхода двоичного дерева. Например: "правый — корень — левый" (симметричный обход):

- обойти правое поддерево
- посетить корень
- обойти левое поддерево

Как видим, это рекурсивные алгоритм. Он должн заканчиваться без повторного вызова, когда текущий корень — пустое дерево. При выводе элементов, обход будем сочетать с выводом отступа '\t' - n раз:

In [61]:

```
def OutTree(x,n):
    if x:
        OutTree( x['right'], n+1 )
        print( '\t'*n,x['val'] )
        OutTree( x['left'], n+1 )
OutTree(tree,0)
```

```
        9
    8
        7
6
        4
    3
        1
```

Деревья поиска

- слева от каждого узла находятся узлы, ключи которых меньше или равны ключу данного узла;
- справа от каждого узла находятся узлы, ключи которых больше или равны ключу данного узла.
-

Дерево сбалансировано - когда для каждой его вершины высота левого и правого поддеревьев различается не более чем на единицу. Если при линейном поиске в массиве за одно сравнение отсекается 1 элемент, здесь — сразу примерно половина оставшихся. Алгоритм поиска имеет асимптотическую сложность
O(log2N)

In [24]:

```
def FindInTree(x,y,n): # x - элемент, y - дерево, n - кол-во шагов
```

```
    if y:
        if x==y['val']:
          print('Элемент',x,'найден за',n,
                n>4 and 'шагов' or
                n>1 and 'шага' or'шаг')
        elif x<y['val']: FindInTree(x,y['left'],n+1)
        else: FindInTree(x,y['right'],n+1)
    else:
      print('Элемент',x,'не найден. Проверено за',n,
                n>4 and 'шагов' or
                n>1 and 'шага' or'шаг')
FindInTree( 8, tree, 0)
FindInTree( 5, tree, 0)
FindInTree( 4, tree, 0)
```

Элемент 8 найден за 1 шаг
Элемент 5 не найден. Проверено за 3 шага
Элемент 4 найден за 2 шага

Вычисление арифметических выражений

Сначала выражение, записанное в линейном виде (в одну строку), нужно "разобрать" и построить соответствующее ему дерево. Затем в результате прохода по этому дереву от листьев к корню вычисляется результат. Пример: 40 - 2 *3 - 4* 5 В корень дерева нужно поместить последнюю из операций с наименьшим приоритетом.

Алгоритм построения дерева:

- найти последнюю выполняемую операцию
- **if** операций нет:
  - создать узел-лист (число)
  - **return**
- поместить найденную операцию в корень дерева
- построить левое поддерево
- построить правое поддерево

Программа получения дерева Tree из строки s:

```
def prt(x): # Функция определения приоритета:
          # "число" - 3; "+-" - 1 или "*/" - 2
    return (x in '+-') and 1 or (x in '*/') and 2 or 3

def last(s): # Функция определения последней операции (минимального приоритета)
          # В какой позиции строки s она находиться
    minPtr=3; k=3
    for i in range(len(s)):
        if prt(s[i])<=minPtr:
            minPtr=prt(s[i])
            k=i
    return k

def makeTree(s):
    y={}
    if any(s):
        k = last(s)
        y['val'] = s[k]
        y['left'] = makeTree( s[ :k] )
        y['right'] = makeTree( s[k+1: ] )
```

```
    return y
s = '40 - 2 * 3 - 4 * 5'.split()
Tree = makeTree(s)
```
OutTree(Tree,0)

```
      5
   *
      4
-
      3
    *
      2
    -
      40
```

Теперь вычислим выражение по дереву. Если в корне находится знак операции, её нужно применить к результатам вычисления поддеревьев:

- n1 = значение левого поддерева
- n2 = значение правого поддерева
- результат = операция (n1, n2)

In [147]:

```
def calcTree(x):
    if x['left']:
        n1 = calcTree(x['left'])
        n2 = calcTree(x['right'])
        op = x['val']
        return op=="+"and n1+n2 or op=="-"and n1-n2 or op=="*"and n1*n2 or n1//n2
    else: return int(x['val'])
calcTree(Tree)
```

Out[147]:

```
14
```

Задания

1. Измените функцию print_tree_inorder так, чтобы она ставила скобки вокруг каждого оператора с парой операндов. Работает ли она корректно и однозначно? Всегда ли необходимы скобки?
2. Напишите лексический анализатор, то есть, функцию, которая принимает строку с выражением и возвращает список лексем.
3. Найдите еще места в функциях, работающих с деревом выражения, где могут возникнуть ошибки, и добавьте соответствующие предложения raise. Протестируйте ваш код с некорректными выражениями.
4. Придумайте различные способы сохранения дерева животных в файл. Реализуйте тот, который считаете самым простым.
5. . Напишите две программы, которые находят все простые числа в диапазоне от 2 до N двумя разными способами: а) проверкой каждого числа из этого диапазона на простоту; б) используя решето Эратосфена. Сравните число шагов цикла (или время работы) этих программ для разных значений N. Постройте для каждого варианта зависимость количества шагов от N, сделайте выводы о сложности алгоритмов.
6. Без использования программы определите, сколько нулей стоит в конце числа 100! 4. Соберите всю программу и вычислите 100!. Сколько цифр входит в это число? 5. Напишите процедуру для ввода длинных чисел из файла в массив (список). 6. Напишите процедуры для сложения и вычитания длинных чисел (не используя

«длинную арифметику» Python). 7. *Напишите процедуры для умножения и деления длинных чисел (не используя «длинную арифметику» Python). 8. Напишите полную программу для извлечения целого квадратного корня из целого числа. Проверьте, как она будет работать, если на вход функции isqrt подать нецелое значение.

Контрольные вопросы

1. Какие преимущества и недостатки имеет алгоритм «решето Эратосфена» в сравнении с проверкой каждого числа на простоту?
2. Что такое «длинные числа» и «длинная арифметика»?
3. В каких случаях необходимо применять «длинную арифметику»?
4. Какое максимальное число можно записать в ячейку размером 64 бита? Рассмотрите варианты хранения чисел со знаком и без знака.
5. Можно ли использовать для хранения длинного числа символьную строку? Оцените достоинства и недостатки такого подхода.
6. Почему неудобно хранить длинное число, записывая первую значащую цифру в начало массива?
7. Почему неэкономично хранить по одной цифре в каждом элементе массива?
8. Сколько разрядов десятичной записи числа можно хранить в одной 16-битной ячейке?
9. Объясните, какие проблемы возникают при выводе длинного числа. Как их можно решать?
10. *Предложите способ вывода «длинного» числа без использования возможностей функции format.
11. Объясните алгоритм поиска целого квадратного корня из числа с помощью метода Герона.
12. *Предложите алгоритм поиска целого кубического корня их числа, основанный на аналогичной идее.

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №35

Тема: Вычисление длины кратчайшего пути между вершинами графа (алго¬ритм Дейкстры).

Количество часов на выполнение: 2.

Цель работы: Научиться находить кратчайшие пути от вершины графа к другой.

Вычислять длины кратчайшего пути между вершинами, познакомится с алгоритмом Дейкстры.

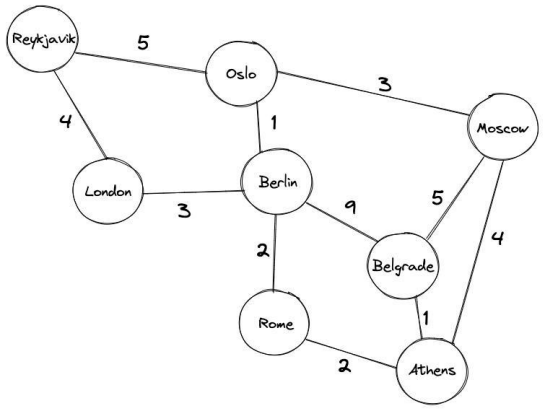
Оборудование: Персональный компьютер, Python, любой редактор разработки (Sublime text, VS code, Notepad, Pycharm).

Методика выполнения задания
Теоретическая часть

Алгоритм Дейкстры — это популярный алгоритм поиска, используемый для определения кратчайшего пути между двумя узлами в графе. В исходном сценарии граф представлял Нидерланды, узлы графа представляли разные голландские города, а рёбра представляли дороги между городами. **Алгоритм Дейкстры можно использовать для решения любой задачи, которая может быть представлена в виде графа**. Предложения друзей в социальных сетях, маршрутизация пакетов через Интернет или поиск пути через лабиринт — все это может сделать алгоритмом Дейкстры. Но как на самом деле это работает?

Напомним, что алгоритм Дейкстры для графов, а это означает, что он может решить проблему, только если она может быть представлена в графоподобном виде. Пример, который будет здесь показан, возможно, самый интуитивно понятный — кратчайший путь между двумя городами.

Мы будем работать с картой, показаной ниже, для поиска налучшего маршрута между двумя европейскими городами Рейкьявиком и Белградом. Для простоты представим, что все города связаны дорогами (реальный маршрут должен включать как минимум один паром).



Обратите внимание на следующее:

Каждый город — это узел.

Каждая дорога — это ребро.

У каждой дороги есть своя ценность. Это может быть расстояние между городами, плата за проезд или интенсивностью движения. Как правило, мы отдаем предпочтение рёбрам с более низкими значениями. В нашем конкретном случае связанное значение определяется расстоянием между двумя городами.

Вы должны уже заметить, что добраться из Рейкьявика до Белграда напрямую невозможно. Это сделало бы наши упражнения бессмысленными. Но есть несколько путей из Рейкьявика в Белград, которые проходят через другие города:

Reykjavik → Oslo → Berlin → Belgrade

Reykjavik → London → Berlin → Rome → Athens → Belgrade

Reykjavik → London → Berlin → Rome → Athens → Moscow → Belgrade

Каждый из этих путей заканчивается в Белграде, но все они имеют разные ценности. Мы можем использовать алгоритм Дейкстры, чтобы найти путь с наименьшим общим значением.

Прежде чем погрузиться в код, посмотрим на алгоритм Дейкстры с высоты птичьего полета.

Сначала инициализируем алгоритм:

Устанавливаем Рейкьявик в качестве начального узла.

Устанавливаем расстояния между Рейкьявиком и всеми другими городами в ∞ **(бесконечность)**, за исключением расстояния между Рейкьявиком и самим собой, которое принимаем равным **0**.

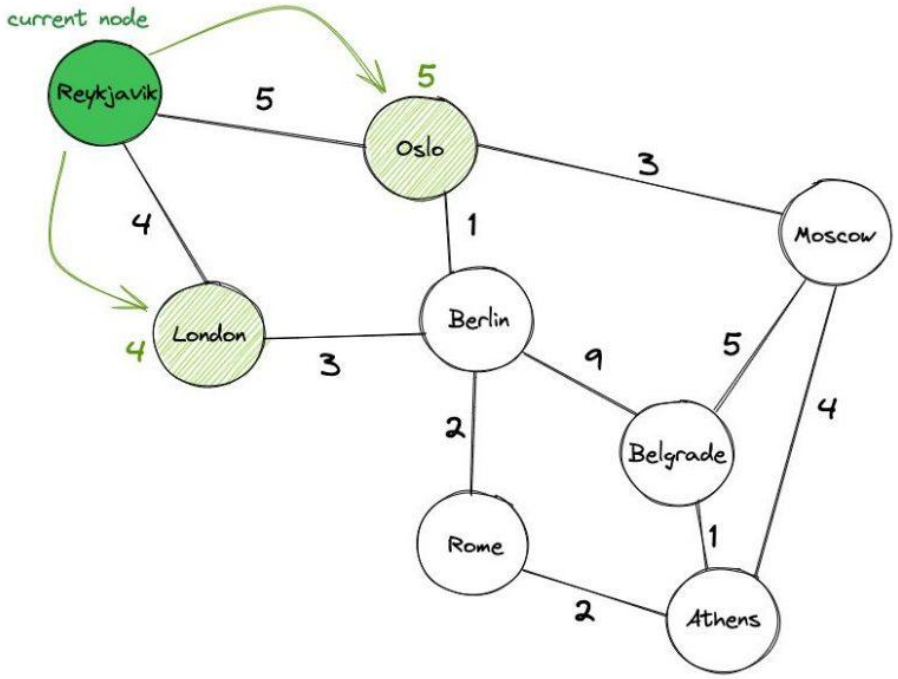
После этого, итеративно выполняем следующие шаги:

Выбираем узел с наименьшим значением в качестве «текущего узла» и посещаем всех соседей. При посещении каждого соседа, обновляем их ориентировочное расстояние от начального узла.

Как только мы посетим всех соседей текущего узла и обновили их расстояния, помечаем текущий узел как **«visited» «посещенный»**). Отметка узла как «посещенного» означает, что найдена его окончательная ценность.

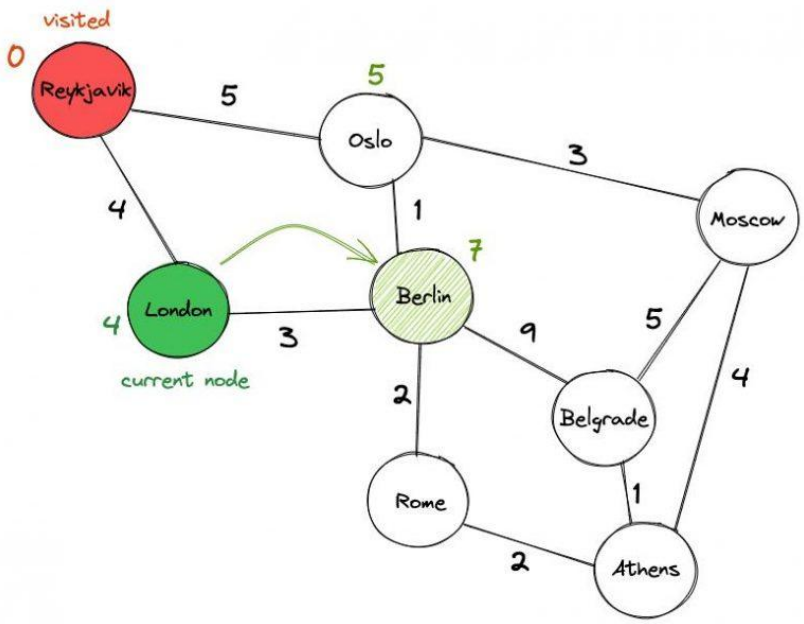
Вернемся к первому шагу и повторяем до тех пор, пока не посетим все узлы графа.

В нашем примере начинаем с отметки Рейкьявика, поскольку его значение равно 0, как «текущего узла». Далее посетим два соседних узла **Рейкьявика**: **Лондон** и **Осло**. В начале алгоритма их значения устанавливаются на бесконечность, но, когда мы посещаем узлы, то обновляем значение для Лондона до 4 и Осло до 5.

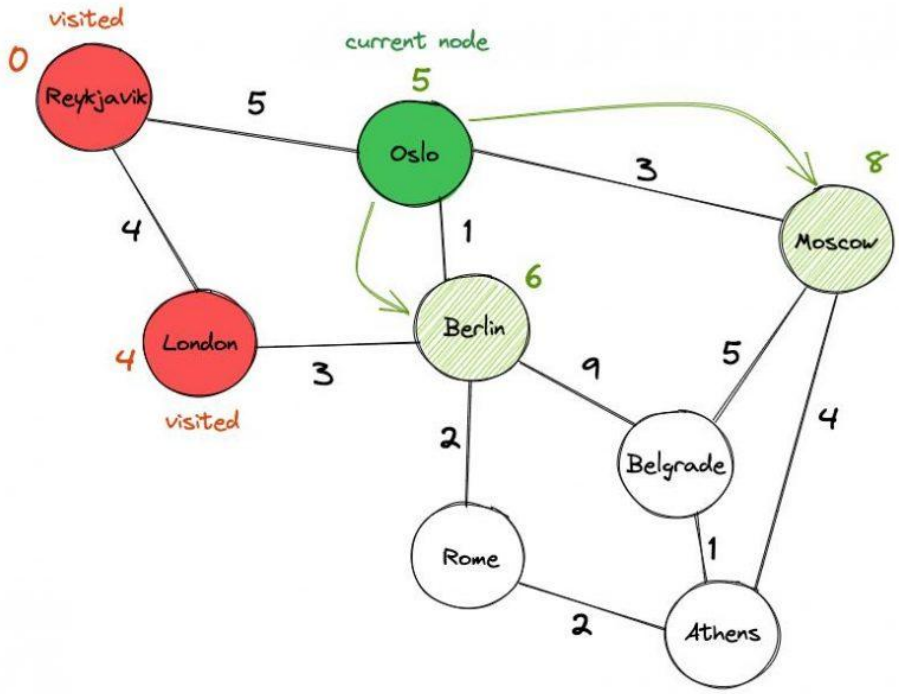


Затем отмечаем Рейкьявик как «посещенный». Мы знаем, что его окончательная стоимость равна нулю, и нам больше не нужно его посещать. Продолжаем со следующего узла с наименьшим значением, которым является **Лондон**.

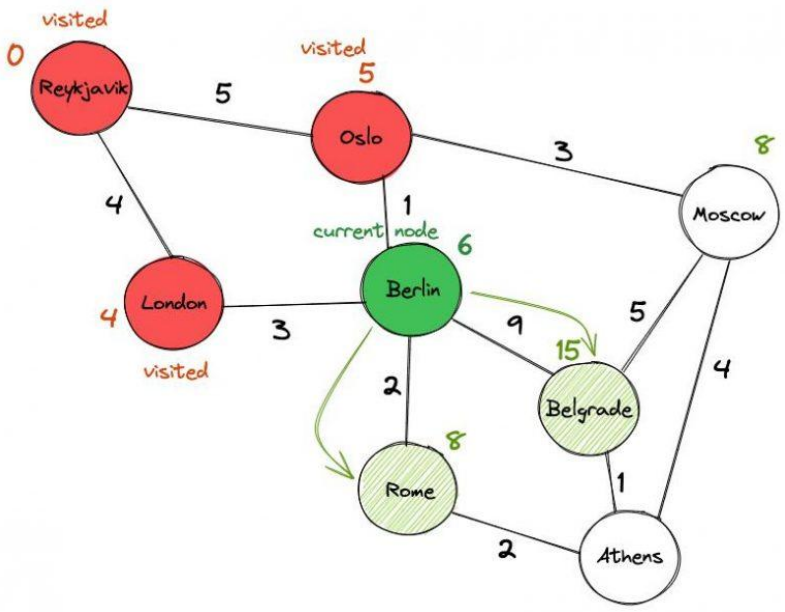
Посещаем все соседние узлы **Лондона**, которые не помечены как «посещенные». Соседи **Лондона** — **Рейкьявик** и **Берлин**. Но мы игнорируем **Рейкьявик**, потому что мы уже были в нем. Вместо этого обновляем значение **Берлина**, добавляя значение ребра, соединяющего **Лондон** и **Берлин** (3), к значению **Лондона** (4), что дает нам значение 7.



Отмечаем **Лондон** как посещенный и выбираем следующий узел — **Осло**. Посещаем соседей **Осло** и обновляем их ценности. Оказывается, можно лучше добраться до **Берлина** через **Осло** (со значением 6), чем через **Лондон**, поэтому соответствующим образом обновляем его значение. Также обновляем текущее значение **Москвы** с бесконечности до 8.



Отмечаем **Осло** как «посещенный» и обновляем его окончательное значение до 5. Между **Берлином** и **Москвой** выбираем **Берлин** в качестве следующего узла, потому что его значение (6) ниже, чем у **Москвы** (8). Действуем так же, как и раньше: посещаем **Рим** и **Белград** и обновляем их предварительные значения, прежде чем пометить **Берлин** как «посещенный» и перейти к следующему городу.

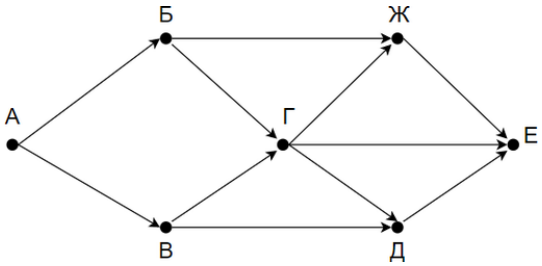


Обратите внимание, что мы уже нашли путь из **Рейкьявика** в **Белград** со значением 15! Но лучший ли он?

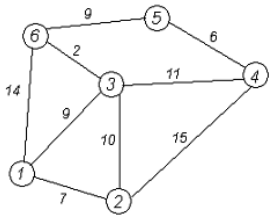
В конце концов, это не так. Пропустим остальные шаги, а для вас это упражнение. Лучшим путем оказывается **Рейкьявик → Осло → Берлин → Рим → Афины → Белград** со значением 11.

Практическая часть

Задание 1. На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город Е?



Задание 2. Требуется найти кратчайшие расстояния от 1 -й вершины до всех остальных для графа, представленного на рисунке:



Задание 3. Между населёнными пунктами A, B, C, D, E построены дороги, протяжённость которых (в километрах) приведена в таблице:

	A	B	C	D	E
A		2		1	
B	2		3	3	
C		3		3	2
D	1	3	3		
E			2		

Определите длину кратчайшего пути между пунктами A и E. Передвигаться можно только по дорогам, протяжённость которых указана в таблице.

Контрольные вопросы:

1. Какой алгоритм поиска на графах находит кратчайшие пути от одной из вершин графа до всех остальных?
2. Когда происходит завершение алгоритма Дейкстры?
3. Как работает алгоритм Дейкстры?
4. Какая сложность у алгоритма Дейкстры?
5. Кто автор первого алгоритма?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №36

Тема: Подсчёт количества вариантов с помощью динамического программирования.

Количество часов на выполнение: 2.

Цель работы: решение задач на Python с помощью динамического программирования.

Оборудование: Персональный компьютер, Python, любой редактор разработки (Sublime text, VS code, Notepad, Pycharm).

Методика выполнения задания

Теоретическая часть

Динамическое программирование – это подход к решению задач оптимизации, когда требуется найти оптимальное решение среди всех возможных подзадач.

Основная идея заключается в том, что решение целой задачи может быть найдено путем решения меньших подзадач, которые могут иметь общие подзадачи. Эти подзадачи берутся из ячеек таблицы, которые заполняются по мере решения меньших задач.

Для использования динамического программирования в Python мы можем создать функцию, которая будет решать каждую подзадачу и заполнять соответствующие ячейки таблицы.

Чтобы успешно решить задачу динамикой нужно:
1) Состояние динамики: параметр(ы), однозначно задающие подзадачу.
2) Значения начальных состояний.
3) Переходы между состояниями: формула пересчёта.
4) Порядок пересчёта.
5) Положение ответа на задачу: иногда это сумма или, например, максимум из значений нескольких состояний.

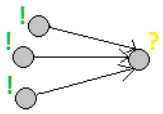
Этапы решения задачи с помощью динамического программирования
Создаём словарь со списком элементов и их параметрами (площадь, ценность).
1. Создаём списки значений площади и ценности.
2. Используем списки для построения таблиц мемоизации.
3. Получаем элементы из последней строки таблицы. Последняя строка таблицы мемоизации содержит оптимальное решение. Возможно, существует несколько оптимальных решений. Например, когда есть два предмета с одинаковой площадью и стоимостью или ряд предметов имеют суммарные площадь и ценность, как у другого предмета.

Мемоизация — (Memoization, англ) вариант кеширования, заключающийся в том, что для функции создаётся таблица результатов, и будучи вычисленной при определённых значениях параметров результат заносится в эту таблицу.
В дальнейшем результат берётся из данной таблицы.
Эта техника позволяет за счёт использования дополнительной памяти ускорить работу программы.

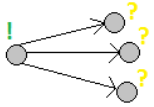
Одномерная динамика

Порядок пересчёта

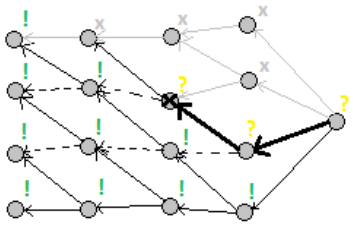
Существует три порядка пересчёта:
1) Прямой порядок:
Состояния последовательно пересчитывается исходя из уже посчитанных.



2) Обратный порядок:
Обновляются все состояния, зависящие от текущего состояния.



3) Ленивая динамика:
Рекурсивная функция пересчёта динамики. Это что-то вроде поиска в глубину по ациклическому графу состояний, где рёбра — это зависимости между ними.



Элементарный пример: числа Фибоначчи. Состояние — номер числа.

Прямой порядок:

```
fib[1] = 1   # Начальные значения
fib[2] = 1   # Начальные значения
for i in range(3, n + 1):
    fib[i] = fib[i - 1] + fib[i - 2]   # Пересчёт состояния i
```

Обратный порядок:

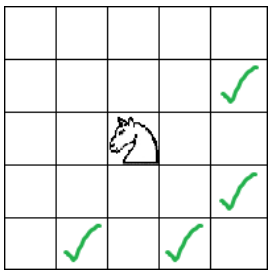
```
fib[1] = 1   # Начальные значения
for i in range(1, n):
    fib[i + 1] += fib[i]   # Обновление состояния i + 1
    fib[i + 2] += fib[i]   # Обновление состояния i + 2
```

Ленивая динамика:

```
def get_fib(i):
    if (i <= 2):   # Начальные значения
        return 1
    if (fib[i] != -1):   # Ленивость
        return fib[i]
    fib[i] = get_fib(i - 1) + get_fib(i - 2)   # Пересчёт
    return fib[i]
```

Все три варианта имеют права на жизнь. Каждый из них имеет свою область применения, хотя часто пересекающуюся с другими.

Многомерная динамика
Многомерная отличается от одномерной количеством измерений, то есть количеством параметров в состоянии. Классификация по этому признаку обычно строится по схеме «один-два-много» и не особо принципиальна, на самом деле.
Многомерная динамика не сильно отличается от одномерной, например:
Пример: Конь
Шахматный конь стоит в клетке `(1, 1)` *на доске размера* `N` x `M`. *Требуется подсчитать количество способов добраться до клетки* `(N, M)` *передвигаясь четырьмя типами шагов:*



Решение:
1) Состояние динамики: dp[i][j] — количество способов добраться до (i, j).
2) Начальное значение: В клетку (1, 1) можно добраться одним способом — ничего не делать.
3) Формула пересчёта:
Для прямого порядка:

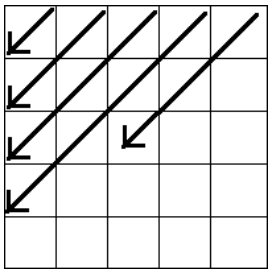
```
dp[i][j] = dp[i - 2][j - 1] + dp[i - 2][j + 1] + dp[i - 1][j - 2] + dp[i + 1][j - 2]
```

Для обратного порядка:

```
dp[i + 1][j + 2] += dp[i][j]
dp[i + 2][j + 1] += dp[i][j]
dp[i - 1][j + 2] += dp[i][j]
dp[i + 2][j - 1] += dp[i][j]
```

4) А теперь самое интересное в этой задаче: порядок. Здесь нельзя просто взять и пройтись по строкам или по столбцам. Потому что иначе мы будем обращаться к ещё не пересчитанным состояниям при прямом порядке, и будем брать ещё недоделанные состояния при обратном подходе.

Есть два пути:
1) Придумать хороший обход.
2) Запустить ленивую динамику, пусть сама разберётся.
Если лень думать — запускаем ленивую динамику, она отлично справится с задачей.
Если не лень, то можно придумать обход наподобие такого:



Этот порядок гарантирует обработанность всех требуемых на каждом шаге клеток при прямом обходе, и обработанность текущего состояния при обратном.
5) Ответ просто лежит в dp[n][m].

Динамика по подотрезкам

Это класс динамики, в котором состояние — это границы подотрезка какого-нибудь массива. Суть в том, чтобы подсчитать ответы для подзадач, основывающихся на всех возможных подотрезках нашего массива. Обычно перебираются они в порядке увеличения длины, и пересчёт основывается, соответственно на более коротких отрезках.

Динамика по поддеревьям

Параметром состояния динамики по поддеревьям обычно бывает вершина, обозначающая поддерево, в котором эта вершина — корень. Для получения значения текущего состояния обычно нужно знать результаты всех своих детей. Чаще всего реализуют лениво — просто пишут поиск в глубину из корня дерева.

Динамика по подмножествам

В динамике по подмножествам обычно в состояние входит маска заданного множества. Перебираются чаще всего в порядке увеличения количества единиц в этой маске и пересчитываются, соответственно, из состояний, меньших по включению. Обычно используется ленивая динамика, чтобы специально не думать о порядке обхода, который иногда бывает не совсем тривиальным.

Динамика по профилю

Классическими задачами, решающимися динамикой по профилю, являются задачи на замощение поля какими-нибудь фигурами. Причём спрашиваться могут разные вещи, например, количество способов замощения или замощение минимальным количеством фигур.

В решениях динамикой обязательно фигурирует состояние — параметры, однозначно задающие подзадачу, но это состояние не обязательно одно единственное. Иногда можно придумать другие параметры и получить с этого выгоду в виде снижения асимптотического времени или памяти.

Подытожим!

Динамическое программирование — популярный метод в компьютерных науках и разработке программного обеспечения, который играет решающую роль в спортивном программировании. Это метод решения сложных проблем путем их разбиения на более мелкие подзадачи и решения каждой подзадачи только один раз с сохранением решений подзадач, чтобы их можно было повторно использовать при необходимости.

Практическая часть

Задача 1. Запишите программу, которая определит количество всех комбинаций монет (1, 5, 10, 25, 50), которыми можно выдать остальные X в автомате по продаже кофе. Задачу решаем с помощью динамического программирования.

Решение:

```python
# Задаем функцию, которая будет вычислять количество комбинаций монет
def count_coin_combinations(coins, target):
    # Создаем массив для хранения количества комбинаций для каждой суммы
    combinations = [0] * (target + 1)
    # Изначально, количество комбинаций с нулевой суммой равно 1
    combinations[0] = 1

    # Для каждой монеты в массиве монет
    for coin in coins:
        # Для каждой суммы от значения монеты до целевого значения
        for amount in range(coin, target + 1):
            # Увеличиваем количество комбинаций для текущей суммы
            # на количество комбинаций для текущей суммы минус значение текущей монеты
            combinations[amount] += combinations[amount - coin]

    # Возвращаем количество комбинаций для целевого значения
    return combinations[target]
```

допишите основную программу, комбинации монет – известны, количество (target) =100.

Задача 2. Собираем рюкзак (knapsack). Имеются n предметов, у каждого из которых есть свой вес (weights) и ценность (values). Нужно выбрать такие предметы, чтобы их суммарный вес не превышал вместимость рюкзака (capacity), а суммарная ценность была максимальной.
(Данный тип задачи известен как задача о рюкзаке).
Решение:

```python
def knapsack(weights, values, capacity):
    n = len(weights)  # Количество предметов
    K = [[0] * (capacity + 1) for _ in range(n + 1)]  # Создаем таблицу размером (n+1) x (capacity+1)

    # Заполняем таблицу
    for i in range(n + 1):
        for w in range(capacity + 1):
            if i == 0 or w == 0:  # Начальные значения - рюкзак пуст, значит значение 0
                K[i][w] = 0
            elif weights[i - 1] <= w:  # Текущий предмет может быть положен в рюкзак
                K[i][w] = max(values[i - 1] + K[i - 1][w - weights[i - 1]],
                              K[i - 1][w])  # Сравниваем ценность текущего предмета плюс ценность
                                            # предметов, которые мы могли бы взять без текущего предмета,
                                            # и ценность предметов без текущего предмета
            else:
                K[i][w] = K[i - 1][w]  # Если текущий предмет нельзя положить в рюкзак, то берем значение из предыдущей строки

    # Восстанавливаем решение
    res = []
    i = n
    w = capacity
    while i > 0 and w > 0:
        if K[i][w] != K[i - 1][w]:
            res.append(i)
            i -= 1
            w -= weights[i]
        else:
            i -= 1

    return K[n][capacity], res[::-1]  # Возвращаем максимальную ценность и список выбранных предметов
```

Допишите программу, если известно:
Вес предметов = [1, 2, 4, 5]
Ценности предметов = [2, 6, 8, 3]
Вместимость = 10
max_value, selected_items = knapsack(weights, values, capacity)
Вывести на экран: максимальную ценность и выбранные предметы

Задача 3. Определение чисел Фибоначчи с использованием динамического программирования, при условии, что n=15

```python
def fib(n): #
    fib_values = [0, 1] #

    if n <= 1: #
        return n

    if len(fib_values) > n: #
        return fib_values[n] #

    for i in range(2, n+1): #
        fib_values.append(fib_values[i-1] + fib_values[i-2]) #

        return fib_values[n]#
```

Расписать комментарии, что означает каждая написанная строчка кода и дописать программу, вывести на экран Fibonacci число с индексом … равно …

Задача 4. Самая длинная общая подпоследовательность

Задача о самой длинной общей подпоследовательности (LCS) — это классическая задача динамического программирования, которая включает в себя поиск самой длинной подпоследовательности, общей для двух заданных строк. Подпоследовательность строки — это последовательность символов, которые появляются в строке в одном и том же порядке, но не обязательно последовательно.

```
def lcs(s1, s2):
    m, n = len(s1), len(s2)
    dp = [[0] * (n+1) for _ in range(m+1)]

    for i in range(1, m+1):
        for j in range(1, n+1):
            if s1[i-1] == s2[j-1]:
                dp[i][j] = dp[i-1][j-1] + 1
            else:
                dp[i][j] = max(dp[i-1][j], dp[i][j-1])

    return dp[m][n]
```

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 37

Тема: Использование растровой графики для хранения фотографий. Конвертация с целью снижения объёма изображения

Количество часов на выполнение: 4, из них на практическую подготовку - 2ч. (при наличии).

Цель работы: Исследовать возможности растровой графики для хранения фотографий, научиться конвертировать изображения в различные форматы с целью снижения объёма файла при сохранении приемлемого качества.

Оборудование: Персональный компьютер, GIMP

Методика выполнения задания
Теоретическая часть

Растровая графика — это тип изображения, созданный из сетки пикселей (квадратных точек). Каждый пиксель содержит информацию о цвете и яркости, что позволяет отображать фотографические изображения с детальной цветовой градацией.

Основные понятия

Пиксель — минимальный элемент изображения в растровых графиках.

Разрешение — число пикселей по горизонтали и вертикали (например, 1920×1080).

Объём файла — размер файла в байтах, зависит от разрешения и формата хранения.

Форматы изображений:

BMP — несжатый формат, занимает много места.

JPEG — сжатие с потерями, уменьшает размер, может ухудшить качество.

PNG — сжатие без потерь, подходит для графики с прозрачностью.

GIF — небольшие анимации и изображения с меньшей цветовой гаммой.

Конвертация и сжатие изображений

Цель: уменьшить размер файла без существенной потери качества.

Методы:

Выбор формата хранения (например, JPEG вместо BMP).

Настройка степени сжатия (качества) при сохранении.

Использование специальных программ и инструментов.

Важные аспекты

Чем выше степень сжатия, тем больше потерь качества.

Для фотографий оптимально использовать JPEG с умеренным уровнем сжатия.

Для изображений с прозрачностью или с малой цветовой палитрой — PNG или GIF.

Практическая часть:

Задание 1: Анализ исходного изображения

Открыть фотографию

Зафиксировать параметры: разрешение, формат файла, размер файла.

Сделать скриншот или записать эти данные.

Задание 2: Конвертация изображения в другой формат с сжатием

Конвертировать исходное изображение в формат JPEG с разными уровнями качества (например, 90%, 70%, 50%).

Сохранить каждую версию и записать полученный размер файла.

Проанализировать, как изменилась величина размера и качество изображения.

Задание 3: Сравнение форматов

Конвертировать исходное изображение в PNG и GIF.

Записать размеры файлов и сравнить их с файлами JPEG.

Сделать вывод о наиболее подходящем формате для хранения фотографии с учетом размера и качества.

Задание 4: Влияние сжатия на качество изображения

Конвертировать изображение в JPEG с очень низким качеством (например, 30%).

Проанализировать визуальные дефекты, если есть.

Сделать вывод о допустимых значениях сжатия для фотографий.

Контрольные вопросы

1. Что такое растровая графика и из каких элементов она состоит?
2. Какие основные форматы растровых изображений существуют и в чем их особенности?
3. Почему важно управлять степенью сжатия изображений?
4. Чем отличается сжатие с потерями от сжатия без потерь?
5. В каких случаях предпочтительно использовать формат JPEG, а в каких — PNG?
6. Какие параметры влияют на занимаемый размер файла растрового изображения?
7. Как снизить объем файла изображения, не ухудшая его качество существенно?
8. Какие потенциальные потери качества могут возникнуть при сильном сжатии изображения?
9. Почему для хранения фотографий часто используют формат JPEG?
10. Какие преимущества и недостатки имеют форматы GIF и PNG для хранения фотографий?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №38

Тема: Интерфейс и настройка его частей. Однооконный и многооконный режим

Количество часов на выполнение: 4.

Цель работы: Ознакомиться с интерфейсом графического редактора GIMP. Научиться переключаться между однооконным и многооконным режимами. Освоить настройку и индивидуализацию интерфейса для удобной работы. Познакомиться с основными элементами управления и инструментами GIMP.

Оборудование: Персональный компьютер, GIMP

Методика выполнения задания

Теоретическая часть

GIMP (GNU Image Manipulation Program) — это свободный и открытый графический редактор, предназначенный для обработки растровых изображений. Он широко используется для редактирования фотографий, создания графики, дизайна и иллюстраций.

Основные режимы интерфейса

1. Однооконный режим

В этом режиме все окна инструментов, панелей и изображений объединены в один основной экран.

Такой режим удобен для комфортной работы, поскольку все элементы находятся в одном окне и их легко отображать и скрывать.

2. Многооконный режим

В этом режиме окна инструментов, диалоговых панелей и изображений разделены на отдельные окна.

Подходит для профессиональной работы и для тех, кто привык к такому стилю, как в старых версиях GIMP или других редакторах.

Настройка интерфейса

А. Переключение режимов

В меню «Windows» (Окна) есть разделы «Single-Window Mode» (Однооконный режим) и «MultWindow Mode» (Многооконный режим).

Для активации одного режима достаточно поставить галочку напротив «Single-Window Mode».

Б. Настройка элементов интерфейса

Можно закреплять, скрывать или перемещать панели инструментов, диалоговые окна (слои, история, инструменты).

Панели можно закреплять слева, справа, сверху или отключать.

В. Сохранение настроек

После изменения вида интерфейса их можно сохранить, чтобы при следующем запуске они автоматически применялись.

Обзор элементов интерфейса

Меню (Файл, Правка, Выделение, Инструменты, Диалоги)

Основное окно редактора — место отображения и редактирования изображения

Панель инструментов — инструменты рисования, выделения, трансформации и др.

Диалоговые окна — слои, история, цвета, кисти, шрифты и др.

Практическая часть

Задание 1: Переключение режимов интерфейса

Открыть GIMP.

В меню «Windows» выбрать «Single-Window Mode».

Сделать скриншот интерфейса.

Повторить для отключения этого режима («Multi-Window Mode») и сделать еще один скриншот.

Описать отличие между двумя режимами.

Задание 2: Настройка интерфейса для удобства

Закрепить панель слоёв справа или слева.

Открепить и переместить окно инструментов в левый или правый край.

Скрыть ненужные диалоговые окна (например, история или шрифты).

Сохранить настройки (если есть возможность).

Задание 3: Пользовательская настройка и сохранение

Настроить интерфейс по своему удобству.

Сохранить текущий интерфейс как пользовательский профиль.

Перезапустить GIMP и проверить сохранение настроек.

Иллюстрации и скриншоты

Скриншот меню «Windows» с отмеченными пунктами «Single-Window Mode» и «Multi-Window Mode».

Скриншот интерфейса в однократном режиме, где все панели объединены.

Скриншот интерфейса в многооконном режиме с раздельными окнами инструментов и слоёв.

Скриншоты отдельных элементов интерфейса при разной настройке.

Контрольные вопросы

1. Что такое однокнопочный режим в GIMP и чем он отличается от многооконного?
2. Каким образом можно переключиться между однококонным и многооконным режимами?
3. Какие преимущества у однококонного режима? А у многооконного?
4. Назовите основные элементы интерфейса GIMP, с которыми вы познакомились.
5. Как закрепить или скрыть панели инструментов и диалоговые окна в GIMP?
6. Каким образом можно сохранить настройки интерфейса для дальнейшего использования?
7. Что делает меню «Windows» в GIMP? Какие пункты оно содержит, относящиеся к режимам интерфейса?
8. Какой режим рекомендуется использовать начинающим для комфортной работы и почему?
9. Когда целесообразно использовать многооконный режим?
10. Как можно настроить расположение элементов интерфейса так, чтобы они не мешали работе?
11. Объясните, почему важно уметь переключаться между режимами интерфейса GIMP.
12. Что нужно сделать, чтобы при запуске GIMP автоматически открывать определённую настройку интерфейса?
13. Как можно временно скрыть или разгрузить интерфейс для быстрой работы с изображением?
14. Из каких элементов состоит окно редактора GIMP в одноконном режиме?
15. Что бы вы порекомендовали новичкам — использовать одноконный или многооконный режим и почему?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №39

Тема: Управление диалогами. Окно слоёв изображения

Количество часов на выполнение: 4.

Цель работы: Освоить работу с диалоговыми окнами в GIMP, особенно с окном слоёв. Научиться управлять окнами, закреплять, скрывать, перемещать и сохранять расположение диалогов. Понять назначение и функции окна слоёв в процессе редактирования изображений.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания

Теоретическая часть

Окно слоёв: назначение и функции

Окно слоёв отображает список всех текущих слоёв изображения.

Позволяет управлять порядком слоёв, их прозрачностью, режимами наложения и видимостью.

Необходимое средство для многослойного редактирования и композиции.

Основные функции окна слоёв

Добавление новых слоёв

Удаление слоёв

Переупорядочивание слоёв (перетаскиванием)

Активация слоя для редактирования

Изменение прозрачности слоя (ползунок или свойство)

Настройка режима наложения (например, «Умножение», «Экран»)

Включение или отключение видимости слоя (глазик слева)

Управление диалогами в GIMP

Окна диалогов могут быть закреплены или отключены.

Их можно перемещать, закрывать, открывать заново через меню «Windows» (Окна).

Можно создавать группировки окон, закреплять их слева, справа или сверху.

Важное свойство — возможность сохранения пользовательских конфигураций интерфейса.

Практическая часть

Задание 1: Открытие и настройка окна слоёв

Откройте любое изображение или создайте новое.

Через меню Windows → Dockable Dialogs → Layers (Окна → Связные диалоги → Слои) откройте окно слоёв.

Перетащите окно слоёв в левую или правую часть интерфейса так, чтобы оно было закреплено.

Сделайте скриншот текущего расположения окна слоёв.

Задание 2: Управление слоями

Создайте новый слой (Layer → New from Visible или через иконку «Создать новый слой»).

Поменяйте порядок слоёв, перетаскивая их внутри окна.

Включите или отключите видимость слоя, щёлкая по глазку слева.

Измените прозрачность слоя с помощью ползунка или диалогового окна.

Попробуйте изменить режим наложения слоя (если доступен).

Сделайте скриншоты каждого этапа с пояснениями.

Задание 3: Работа с диалогами

Откройте другие диалоги, например, «Цвета» или «История» через меню «Windows → Dockable Dialogs».

Попытайтесь закрепить эти окна рядом с окном слоёв.

Попробуйте временно скрыть диалог и вернуть его обратно.

Попробуйте удалить или закрыть ненужное диалоговое окно.

Сохраните текущую конфигурацию интерфейса (если такая возможность есть).

Задание 4: Повторное восприятие интерфейса

Перезапустите GIMP.

Проверьте, сохраняется ли расположение диалоговых окон.

Настройте интерфейс по своему усмотрению и зафиксируйте конфигурацию, чтобы потом быстро возвращаться к ней.

Важные советы и рекомендации

Для быстрого доступа к управлению слоями используйте сочетания клавиш (например, Ctrl + L– открыть окно слоёв).

Не бойтесь экспериментировать с расположением и скрытием диалогов — это повысит вашу эффективность.

После разметки интерфейса рекомендуется сохранять его конфигурацию (если есть такая опция) для быстрого восстановления при следующем запуске.

Контрольные вопросы

1. Как открыть окно слоёв в GIMP, если оно не отображается?
2. Какие действия можно выполнять с помощью окна слоёв? Назовите хотя бы три.
3. Как изменить порядок слоёв в окне слоёв?
4. Что происходит при отключении видимости слоя в окне слоёв?
5. Как изменить прозрачность выбранного слоя?
6. Какие режимы наложения слоёв вы знаете и для чего они используются?
7. Чем отличается закреплённое диалоговое окно от плавающего?
8. Как можно закрепить окно слоёв в интерфейсе GIMP?
9. Какие ещё диалоговые окна, кроме слоёв, могут быть полезны при работе с изображениями?
10. Как открыть или закрыть диалоговое окно через меню «Windows»?
11. Можно ли одновременно отображать несколько диалоговых окон? Как это сделать?
12. Как сохранить настройки интерфейса с текущими диалогами для дальнейшего использования?
13. Для чего полезна группировка диалогов?
14. Как временно скрыть диалоговое окно и вернуть его обратно?
15. Какие преимущества даёт грамотное управление диалогами при работе с многослойными изображениями?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №40

Тема: Масштабирование и кадрирование: основы работы с изображениями в GIMP.

Количество часов на выполнение: 4.

Цель работы: Научиться изменять размер изображения (масштабировать). Освоить инструменты кадрирования изображения. Понять, как управлять областью просмотра и подготовить изображение к дальнейшему редактированию или сохранению.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания

Теоретическая часть

Основные понятия

Масштабирование изображения

Процесс изменения размера изображения или его части.

Используется для уменьшения или увеличения изображения для удобства работы или итоговой печати.

В GIMP есть несколько способов масштабировать изображение:

Меню Image → Scale Image (Изображение → Масштабировать изображение)

Инструмент «Масштаб» (Scale Tool)

Кадрирование изображения

Процесс обрезки или выделения части изображения с целью получения нужного фрагмента.

Используется для удаления лишних областей и фокусировки на важной части.

В GIMP используется инструмент «Выделение по прямой», «Рамка» или «Инструмент кадрирования»».

Основные операции

Масштабирование

Выберите меню Image → Scale Image.

В появившемся диалоге укажите новые размеры по ширине и высоте.

Выберите единицы измерения (пиксели, проценты, дюймы).

Можно сохранить пропорции, поставив галочку «Цепочка» между полями ширины и высоты.

Нажмите «Масштабировать» для применения.

Инструмент «Масштаб»

В панели инструментов выберите Scale Tool.

Кликните по изображению, чтобы выбрать область масштабирования.

Перетягивайте углы или стороны для изменения размера выделенной части.

Подтвердите изменения.

Кадрирование

Выберите инструмент Кадрирование (Crop Tool).

Обведите область, которую хотите оставить.

Дважды кликните внутри области или нажмите Enter, чтобы применить кадрирование.

Лишние части изображения исчезнут.

Советы и рекомендации

Перед масштабированием рекомендуется сохранить исходное изображение.

При значительном увеличении могут появиться артефакты или потеря качества.

Используйте «Отмену» (Ctrl + Z) для быстрого исправления ошибок.

Для точной кадрировки используйте параметры фрейма или сетку, чтобы получить ровный и профессиональный результат.

Практическая часть

Задание 1. Масштабирование всего изображения

Откройте любое цветное изображение в GIMP.

Сохраните исходное изображение под другим именем (например, с добавлением "_копия").

Через меню Image → Scale Image уменьшите ширину изображения на 50% (с сохранением пропорций).

Просмотрите результат, оцените изменение качества.

Сделайте скриншот результата.

Задание 2. Масштабирование части изображения

Откройте изображение.

Выделите прямоугольную область с помощью инструмента «Выделение по прямоугольной области».

Воспользуйтесь инструментом Scale Tool для масштабирования выделенной области (увеличьте или уменьшите её).

Подтвердите изменения, сохранив пропорции.

Сравните изменения с исходным изображением. Сделайте скриншоты до и после.

Задание 3. Кадрирование изображения

Откройте любое изображение.

Выберите инструмент Crop Tool (Кадрирование).

Выделите в изображении область, которую хотите оставить (например, вырежьте только центральную часть).

Примените кадрирование.

Сохраните полученное изображение под новым именем.

Задание 4. Кадрирование с точными размерами

Выберите инструмент Crop Tool.

В параметрах инструмента задайте точные размеры области кадрирования (например, 800×600 пикселей).

Вырежьте область с этими размерами из изображения.

Сохраните результат и сделайте сравнительный анализ с исходным изображением.

Задание 5. Комбинированная работа с масштабированием и кадрированием

Откройте изображение.

Сначала масштабируйте изображение, увеличив ширину на 120% с сохранением пропорций.

После этого вырежьте с помощью инструмента кадрирования непосредственно масштабированную часть, оставив нужный объект в центре.

Сохраните результат.

Задание 6. Проверка работы с отменой действий

Выполните любое масштабирование или кадрирование изображения.

Отмените действие с помощью Ctrl + Z.

Повторите масштабирование или кадрирование несколько раз с разными параметрами.

Проанализируйте, как влияет отмена на процесс редактирования.

Контрольные вопросы

1. Какими способами можно масштабировать изображение в GIMP?
2. Что значит «сохранять пропорции» при масштабировании, и зачем это нужно?
3. Как изменить размер изображения в пикселях?
4. Какой инструмент используется для кадрирования изображения в GIMP?

5. Что произойдет, если вы масштабируете изображение без сохранения пропорций?

6. Почему важно сохранять исходное изображение перед масштабированием?

7. Как отменить ошибочное масштабирование или кадрирование?

8. Можно ли изменять только часть изображения (например, выделенный фрагмент), или всегда масштабируется всё изображение целиком?

9. Как подготовить изображение к печати с нужным масштабом?

10. Чем отличается масштабирование по размерам и кадрирование?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа № 41

Тема: Заливка, фильтры и инструменты рисования: основы работы с изображениями в GIMP.

Количество часов на выполнение: 4.

Цель работы: Освоить инструменты заливки и рисования в GIMP. Познакомиться с основными фильтрами для обработки изображений. Научиться применять эффекты для улучшения и стилизации изображений.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания

Теоретическая часть

Теоретическая концепция

Инструменты рисования

Кисть (Paintbrush): стандартный инструмент для ручного рисования.

Аэрограф (Airbrush): имитирует распыление краски.

Карандаш (Pencil): рисование без мягкости краев.

Ластик (Eraser): удаляет части изображения.

Инструмент заливки (Bucket Fill Tool): закрашивает выбранную область или весь слой цветом или узором.

Фильтры

В GIMP доступны разнообразные фильтры для искажения, размытия, повышения резкости, создания художественных эффектов.

Например, Размытие (Blur), Резкость (Sharpen), Фильтры художественной обработки.

Практическая часть

Задание 1. Заливка области цветом или узором

Откройте новое изображение или используйте пустой холст.

Выберите инструмент Bucket Fill Tool.

В параметрах выберите цвет или узор.

Кликните по области, чтобы залить выбранной заливкой.

Попробуйте залить всю картинку целиком и отдельные части.

Задание 2. Рисование простым кистевым инструментом

Выберите инструмент Кисть.

В настройках выберите тип кисти, размер и мягкость.

Нарисуйте линии, фигуры или произвольный рисунок на слое.

Измените цвет кисти и повторите.

Задание 3. Создание художественного эффекта с помощью фильтров

Импортируйте или откройте изображение.

Примените фильтр Размытие → Гаусс для смягчения деталей.

Используйте фильтр Резкость → Умная резкость.

Попробуйте эффект Масляной живописи или Картина (если есть).

Оцените результат и сравните с исходным изображением.

Задание 4. Использование инструментов рисования с эффектами

Нарисуйте линию или фигуру с помощью Карандаша.

Добавьте тень или световой эффект через фильтр или градиенты.

Попробуйте применить фильтр Искажение → Волна для создания эффекта искажения.

Задание 5. Создание композиции

Нарисуйте фон с помощью заливки или градиента.

Добавьте основные объекты с помощью кистей.

Примените эффекты и фильтры для стилизации композиции.

Контрольные вопросы

1. Какие инструменты рисования доступны в GIMP, и для чего каждый из них предназначен?
2. Чем отличается инструмент заливки от инструмента градиента?
3. Как применить фильтр для размытия изображения? Назовите хотя бы один.
4. Что такое «маска слоя» и как она связана с применением фильтров?
5. Как изменить параметры кисти (размер, мягкость, жесткость)?
6. Какие виды фильтров помогают повысить резкость изображения?
7. Как можно объединить рисование и фильтры для обработки изображения?
8. Чем отличается применение фильтра к слою и к изображению в целом?
9. Какие настройки нужно менять, чтобы добиться эффекта «имитации масляной живописи»?
10. Почему важно сохранять промежуточные версии работы?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная
Ссылки на источники:*[1]*

Практическая работа №42

Тема: Выделение, контуры, комбинирование изображений: основы работы с изображениями в GIMP.

Количество часов на выполнение: 4.

Цель работы: Научиться выделять сложные объекты и фон. Освоить создание и редактирование контуров. Научиться объединять несколько изображений в одну композицию.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания
Теоретическая часть

Выделение
Процесс отделения части изображения для её редактирования.
Основные инструменты:
Выделение прямоугольной или эллиптической области
Магнитное выделение
Инструмент «Лассо» (Free Select Tool)
Инструмент «Зоны» (Fuzzy Select) — волшебная палочка
Инструмент «Выделение по цвету»
Контуры
Создание путей (лафет), которые могут быть использованы для точного вырезания объектов.
Инструменты:
Перо (Paths Tool)
Создание замкнутых contours (контуров) для точного выделения и последующего редактирования.
Комбинирование изображений
Использование слоёв, масок и режимов наложения для объединения нескольких изображений.
Методы:
Копирование и вставка слоёв
Маски слоя для управляемого объединения
Регулировка прозрачности и режимов наложения (Overlay, Multiply и др.)

Практическая часть

Задание 1. Выделение сложной фигуры
Откройте изображение с простым объектом (например, дерево, человек).
Используйте Инструмент «Лассо» для произвольного выделения объекта.
Тонко подгоняйте выделение, используя Магнитное выделение или Зоны по цвету для более точной отделки.
Создайте новый слой с выделенной областью: нажмите Правой кнопкой — Выделение → Вырезать и вставьте как новый слой.
Задание 2. Создание и редактирование путей (контуров)

Используя Инструмент «Перо» (Paths Tool), нарисуйте контур вокруг объекта, который хотите вырезать.

Замкните путь, щёлкнув по начальной точке.

В меню «Пути» преобразуйте контур в выделение (выберите — «Образовать выделение»).

Вырежьте выделенную область или скопируйте её.

Задание 3. Комбинирование двух изображений

Откройте два изображения с разными объектами.

Скопируйте один слой и вставьте его в другое изображение (через Правка → Вставить).

Используйте маску слоя для скрытия ненужных частей, создавайте мягкие границы с помощью мягкой кисти или градиента.

Регулируйте режимы наложения слоя, чтобы добиться гармоничного сочетания.

Сделайте итоговую композицию, объедините слои и сохраните.

Задание 4. Использование маски слоя

Создайте копию слоя.

Добавьте маску слоя к копии (через Правка → Добавить маску слоя).

Нарисуйте черным цветом по маске — скрывая части слоя, белым — показывая.

Используйте软кую кисть для мягких масок.

Визуально объедините слои с помощью маски.

Контрольные вопросы

Какие инструменты выделения наиболее подходят для сложных фигур? В чем их особенности?

Как создать и использовать контуры (пути) для точного выделения?

Чем отличается выделение с помощью волшебной палочки и «Лассо»?

Что такое маска слоя и как она помогает при объединении изображений?

Как совместить несколько изображений в одну композицию?

Почему важно работать с слоями при комбинировании изображений?

Для чего используют режимы наложения слоёв (например, Overlay, Multiply)?

Какие способы улучшить границы при объединении изображений?

Как легко исправить ошибку выделения или маски?

Какие методы позволяют добиться реалистичного комбинирования изображений?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №43

Тема: Создание эффектов с использованием быстрой маски и цветовых преобразований в GIMP

Количество часов на выполнение: 4.

Цель работы: Освоить работу с быстрой маской для точного выделения областей. Познакомиться с основными цветовыми преобразованиями. Научиться создавать художественные и коррекционные эффекты с помощью цветокоррекции и масок.

Оборудование: Персональный компьютер, Gipm

Методика выполнения задания
Теоретическая часть

Быстрая маска (Quick Mask)

Временный режим выделения, который отображается как цветная прозрачная маска поверх изображения.

Позволяет редактировать выделение кистью и другими инструментами рисования.

Активируется кнопкой внизу панели выделения или клавишей Shift + Q.

Цветовой оттенок (обычно красный) показывает невидимые области, а прозрачные — выделенные.

Цветовые преобразования

Применение различных операций с цветами для улучшения или изменения изображения.

Основные инструменты:

Яркость и контраст

Уровни (Levels) — позволяет настраивать тональный диапазон

Кривые (Curves) — более точная коррекция яркости и цвета

Оттенок и насыщенность (Hue-Saturation)

Цветовой баланс (Color Balance)

Инверсия цвета (Invert)

Создание эффектов

Использование быстрой маски для точного выделения элементов.

Цветовые преобразования применяются только к выделенным областям.

Таким образом можно создавать эффекты подсветки, цветового смещения, выделения отдельных деталей.

Практическая часть

Задание 1. Работа с быстрой маской для выделения объекта

Откройте любое изображение с четким объектом.

Активируйте быструю маску (Shift + Q).

Используйте кисть (с черным цветом) чтобы добавить к маске области, которые хотите исключить из выделения.

Кистью белого цвета удалите лишнее из маски.

Выключите быструю маску (Shift + Q), выделение появится на основании отредактированной маски.

Скопируйте выделенный объект на новый слой.

Задание 2. Применение цветовых преобразований к выделенной области

Сделайте выделение любого объекта на изображении (через быструю маску или другие инструменты).

Перейдите в меню Цвета → выберите, например, Оттенок и насыщенность.

Измените оттенок, насыщенность или яркость для выделенного объекта.

Поэкспериментируйте с другими цветовыми коррекциями — Уровни, Кривые, Цветовой баланс.

Сравните эффект до и после.

Задание 3. Создание эффекта цветового выделения объекта («Колоризация»)

Откройте цветное изображение.

Создайте выделение объекта с помощью быстрой маски.

Инвертируйте выделение (Выделение → Инвертировать).

Примените Оттенок и насыщенность к фону и полностью обесцветьте фон (сдвиньте насыщенность в -100).

Оставьте объект цветным, в результате выделится цветная часть на черно-белом фоне.

Задание 4. Создание эффекта свечения с помощью быстрой маски

Выделите объект с помощью быстрой маски.

Создайте копию выделенного объекта на новом слое.

Примените фильтр Размытие → Гауссово размытие к копии для создания «светящегося» ореола.

Настройте прозрачность слоя с размытием для нужного эффекта свечения.

Контрольные вопросы

1. Что такое быстрая маска и как она помогает в выделении областей?
2. Как переключить режим быстрой маски в GIMP?
3. Чем отличаются цветовые преобразования «Уровни» и «Кривые»?
4. Как сделать цветной объект на черно-белом фоне с помощью выделения?
5. Какие инструменты в GIMP используются для изменения оттенка и насыщенности изображения?
6. Что происходит, если наносить кистью черный цвет на быстрой маске? А белый?
7. Как можно использовать быструю маску для создания эффекта свечения?
8. Для чего нужен инверт выделения при работе с цветовыми эффектами?
9. Почему удобнее применять цветокоррекцию именно к выделенным областям, а не к всему изображению?
10. Как сохранять промежуточные результаты работы с масками и цветовыми преобразованиями?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №44

Тема: Градиенты в GIMP: от создания до применения в графических работах

Количество часов на выполнение: 4.

Цель работы: Освоить создание, настройку и редактирование градиентов. Научиться применять градиенты для оформления фонов, элементов и эффектов. Развить навыки использования градиентов в комплексных графических работах.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания

Теоретическая часть

Что такое градиент?

Плавной переход между двумя или несколькими цветами.

Может использоваться для фона, заливки, создания эффектов свечения, теней, объемности.

Типы градиентов в GIMP

Линейный — цветовой переход вдоль прямой линии.

Круговой — цветовой переход по радиусу окружности.

Радиальный — аналогично круговому, расширяется от центра.

Конус, сферический, затухающий и др. — дополнительные типы в настройках.

Создание и настройка градиента

Можно создавать собственные градиенты через редактор градиентов.

В редакторе можно добавлять, удалять или менять цветовые точки (узлы).

Использовать встроенные и пользовательские градиенты.

Практическая часть

Задание 1. Создание собственного градиента

В панели инструментов выберите Инструмент «Заливка градиентом» (Gradient Tool).

В свойствах инструмента (панель опций) нажмите на кнопку Редактор градиентов.

В открывшемся окне нажмите Новый или Дублировать.

Настройте новые цветовые точки: выберите цвет, позицию (от 0 до 1).

Добавьте несколько точек для плавных переходов или эффекта радужности.

Сохраните созданный градиент для использования в работе.

Задание 2. Применение градиента к фону

Создайте новый документ или откройте существующее изображение.

Выберите Инструмент «Заливка градиентом».

В свойствах выберите созданный вами градиент.

Потяните мышь по изображению, чтобы залить его градиентом (например, от верхнего левого угла к нижнему правому).

Попробуйте разные направления и длины линий заливки.

Задание 3. Использование градиента для оформления элементов

Нарисуйте или выберите фигуру (например, круг или прямоугольник).

Вырежьте её или создайте на отдельном слое.

Используйте Инструмент «Заливка градиентом» и примените градиент внутри фигуры.

Для более эффектного результата, примените маску слоя или слой с градиентом с режимами наложения (например, Overlay).

Задание 4. Создание эффектов с градиентами

Нарисуйте абстрактный фон или объект.

Используйте градиенты с прозрачностью (по настройкам градиента) для создания свечений, тумана или мягких теней.

Попробуйте добавлять градиенты поверх уже нарисованных элементов для усиления объема.

Задание 5. Анимация с градиентами (при наличии навыков)

Можно комбинировать градиенты с масками и слоями для создания эффектов движения или переливов.

Контрольные вопросы

1. Чем отличается линейный градиент от радиального? Назовите ситуации их использования.
2. Как создать собственный градиент в GIMP? Какие параметры можно изменять?
3. Каким образом можно сделать градиент прозрачным?
4. Почему важно сохранять созданные градиенты для повторного использования?
5. Как связать градиент с прозрачностью для создания эффектов свечения?
6. Какие режимы наложения слоёв лучше использовать при работе с градиентами?
7. В чем преимущество использования градиентов при создании фонов?
8. Как можно комбинировать градиенты с текстом или фигурами?
9. Какие инструменты лучше всего подходят для точного применения градиентов?
10. Какие идеи для использования градиентов в графическом дизайне вы можете придумать?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Практическая работа №45

Тема: Создание анимированных GIF в GIMP

Количество часов на выполнение: 4.

Цель работы: Научиться создавать анимационные GIF-файлы, используя слои и временную шкалу GIMP. Освоить подготовку кадров, настройку времени отображения и экспорт анимации.

Оборудование: Персональный компьютер, Gimp

Методика выполнения задания

Теоретическая часть

Что такое анимационный GIF?

Формат графического файла, состоящий из серии изображений (кадров), показываемых по очереди для создания иллюзии движения.

Используется для коротких анимаций, мемов, объясняющих иллюстраций.

Основные принципы в GIMP

Каждый кадр — это отдельный слой.

Последовательность слоёв формирует анимацию.

Время каждого кадра задаётся при экспорте через параметры задержки.

При экспорте выбирается формат GIF с опцией "Анимация".

Практическая часть

Задание 1. Подготовка кадров

Создайте новый документ с нужным размером (например, 500x500 px).

Нарисуйте или импортируйте базовые элементы для анимации (например, движущийся объект).

Дублируйте слой (нажатием правой кнопкой по слою → Дублировать слой) для создания последовательных кадров.

В каждом новом слое измените положение или свойства объекта для создания эффекта движения.

Задание 2. Установка времени отображения кадров

После создания слоёв убедитесь, что их порядок правильный — первый кадр сверху или снизу (зависит от версии GIMP).

Назначьте задержку для каждого слоя:

Правый клик по слою → Настройки слоя → Вводите задержку в миллисекундах в начале имени слоя, например, Layer name (100ms).

Многие версии GIMP позволяют задавать задержку при экспорте без ручной маркировки, выбирая её глобально.

Задание 3. Экспорт анимации как GIF

Перейдите в меню Файл → Экспорт как….

Выберите формат GIF (имя_файла.gif).

В диалоговом окне:

Отметьте Как анимацию.

Укажите задержку между кадрами (или используйте задержку, заданную в слоях).

Включите Повторять навсегда для бесконечной анимации.

Нажмите Экспорт.

Задание 4. Просмотр и проверка

Откройте полученный GIF в браузере или просмотрщике изображений, чтобы убедиться в корректности анимации.

При необходимости вернитесь к GIMP, отрегулируйте задержки или содержание кадров, и снова экспортируйте.

Советы и рекомендации

Используйте слои с прозрачностью для сложных эффектов.

Организуйте слои по порядку — сверху самый первый кадр.

Для плавной анимации делайте небольшие изменения между кадрами.

Экспериментируйте с задержками, чтобы добиться нужного ритма.

Варианты использования

Создание мемов и коротких роликов.

Обозначения и анимации для презентаций.

Разработка шевеленки для логотипов и элементов дизайна.

Контрольные вопросы:

1. Что такое анимационный GIF и из каких элементов он состоит?
2. Как в GIMP отмечаются отдельные кадры анимации?
3. Почему каждый слой в GIMP считается отдельным кадром при создании GIF?
4. Как задать время отображения каждого кадра при создании анимации в GIMP?
5. Какие параметры нужно выбрать при экспорте файла для получения анимационного GIF?
6. В чем заключается разница между склейкой кадров и настройкой задержки?
7. Можно ли создавать анимации из неподвижных изображений? Как?
8. Какие советы дасте для создания плавной и эффективной анимации?
9. Какие форматы изображений можно экспортировать из GIMP для последующего использования?
10. В чем преимущества использования GIMP для создания анимационных GIF по сравнению с другими программами?

Требования к оформлению отчетного материала:

Отчет оформляется по специально заданной структуре и предоставляется для оценивания преподавателю.

Содержание отчета:

Отчет должен содержать:

1.Название работы.

2.Цель работы.

3.Результаты выполнения заданий.

4. Ответы на контрольные вопросы.

Практическая работа сохраняется в личной папке на сетевом диске.

Затем прикрепляется в Электронный курс https://el.istu.edu Информатика

Форма контроля: дифференцированная

Ссылки на источники:*[1]*

Методические указания по дисциплине ОУП.05 Информатика составлены в соответствии с рабочей программой.

Составитель:
Шипилова Анастасия Дмитриевна, преподаватель

Методические указания рассмотрены и рекомендованы к утверждению на заседании цикловой комиссии Монтажа и ремонта промышленного оборудования

Протокол № 3 от « 6 » 11 2025 г.
Председатель ЦК _____ Т.В. Данилова

СОГЛАСОВАНО:
Заместитель декана по учебно-производственной работе

_____ П.М. Макогон
« 6 » 11 2025 г.

УТВЕРЖДАЮ:
Заместитель декана
по учебной работе
_____ И.А.Чинская